Gradient-Ascent Pulse Engineering with Feedback

Riccardo Porotti,[1,2,*] Vittorio Peano,[1] and Florian Marquardt[1,2]

[1] *Max Planck Institute for the Science of Light, 91058 Erlangen, Germany*

[2] *Department of Physics, Friedrich-Alexander Universität Erlangen-Nürnberg, 91058 Erlangen, Germany*

(Received 4 April 2022; revised 3 May 2023; accepted 16 June 2023; published 13 July 2023)

Efficient approaches to quantum control and feedback are essential for quantum technologies, from sensing to quantum computation. Open-loop control tasks have been successfully solved using optimization techniques, including methods such as gradient-ascent pulse engineering (GRAPE) , relying on a differentiable model of the quantum dynamics. For feedback tasks, such methods are not directly applicable, since the aim is to discover strategies conditioned on measurement outcomes. In this work, we introduce feedback GRAPE, which borrows some concepts from model-free reinforcement learning to incorporate the response to strong stochastic (discrete or continuous) measurements, while still performing direct gradient ascent through the quantum dynamics. We illustrate its power considering various scenarios based on cavity-QED setups. Our method yields interpretable feedback strategies for state preparation and stabilization in the presence of noise. Our approach could be employed for discovering strategies in a wide range of feedback tasks, from calibration of multiqubit devices to linear-optics quantum computation strategies, quantum enhanced sensing with adaptive measurements, and quantum error correction.

DOI: 10.1103/PRXQuantum.4.030305

I. INTRODUCTION

The application of optimal-control techniques to quantum systems [1,2] forms a cornerstone of modern quantum technologies, ranging from the tailoring of laser pulses acting on molecules to the synthesis of unitaries in multiqubit systems as part of the "compilation" of quantum algorithms for specific hardware platforms. Since the equations of quantum dynamics are explicitly known and even differentiable, one can exploit this knowledge and, specifically, make use of powerful gradient-based techniques. The most prominent approach is "gradient-ascent pulse engineering" (GRAPE) [3,4], with its efficient evaluation of gradients, together with its variants. GRAPE is the state-of-the-art method for quantum optimal control and is extremely widely employed. In fact, it has been used to find optimal control sequences for spin systems [3,5,6], coupled qubits [7,8], an implementation of the Jaynes-Cummings model [9], and qubit-cavity lattices [10], among many other examples. It has also been used to optimize open dynamics [11,12], it has been turned into an adaptive approach to cope with parameter uncertainties [13–16], and it has been extended to second-order optimization techniques [17]. Other efficient gradient-based optimal control approaches have also been presented recently (see, e.g., Ref. [18]).

However, there is one crucial extension that is not easily addressed by such gradient-based techniques: feedback. Conditioning the control sequence based on the stochastic outcomes of quantum measurements is an important component of many more challenging tasks [19]. It allows us to remove entropy from the system and is therefore essential in applications such as state preparation and stabilization in the presence of noise [20–24], adaptive measurements [25], or quantum error correction with its syndrome extraction (see, e.g., Refs. [26–29]). Unfortunately, discovering feedback strategies is a formidable challenge. These strategies live in a space that is combinatorially larger than that of open-loop control strategies, since every sequence of measurement outcomes may require a different response. Beyond that general difficulty, it is unclear *a priori* how to take gradients through strong stochastic quantum measurement events.

In principle, there is a set of techniques from machine learning that can discover feedback strategies without taking gradients through quantum dynamics: so-called model-free reinforcement learning (RL) [30] approaches (for a brief review, see Appendix A). During the past few years, a number of groups have demonstrated numerically the promise of model-free RL for quantum physics. This has included both open-loop control tasks (see, e.g., Refs. [31–34] and also, in an experiment, Ref. [35]) but, in

*riccardo.porotti@mpl.mpg.de

particular, the more challenging quantum real-time feedback tasks that rely on adaptive responses to measurement outcomes [36–39], recently showcased in the first experiments [40,41]. In model-free RL, the quantum device is treated as a black box, which can be an advantage in applications to experimental setups with unknown parameters [35,38,40,41]. On the other hand, much of the training time is therefore spent in learning implicitly a model of the dynamics while simultaneously attempting to find good feedback strategies. This can make learning inefficient, leading to longer training times and/or suboptimal strategies.

It would therefore seem desirable to find a way to incorporate feedback based on arbitrary quantum measurements into a direct gradient-based optimal control technique such as GRAPE, making efficient use of our knowledge of the differentiable quantum dynamics.

In this work, we present such a technique, which we refer to as "feedback GRAPE." In the language of RL, it would be classified as a model-based technique [42]. On the one hand, it retains the ability of GRAPE to exploit gradients through the quantum dynamics, making the learning more efficient. On the the other hand, similar to model-free RL, it provides a flexible approach to incorporate feedback, even in the presence of strong stochastic measurements.

Overall, the technique we introduce here, feedback GRAPE, is conceptually simple: GRAPE-type gradient ascent for the continuous control parts, possibly implemented using automatic differentiation for convenience [43–48] and in any case exploiting modern gradient optimizers, supplemented with Monte Carlo sampling of measurement outcomes.

We show that the introduction of stochastic sampling in the framework of gradient-based feedback optimization requires the addition of an important correction term to the overall reward function, for discrete measurement outcomes, or a "reparametrization" of the measurement probability density, for continuous measurement outcomes. This innovation allows us to optimize any differentiable reward over long sequences of measurements, both discrete and continuous.

In this way, feedback GRAPE is able to go significantly beyond existing gradient-based optimization methods for feedback. These are typically limited to greedy optimization over one or, at most, a few measurements [21]. Otherwise, for nongreedy optimization, they are limited to optimal control problems that can be mapped onto so-called classical linear-quadratic-Gaussian control problems or to a special linear ansatz for the feedback protocol, as in so-called Markovian quantum feedback methods [19,49]. The special limiting case of weak Gaussian-distributed measurements, which does not yet require the mathematical treatment that we introduce, has recently been considered in Ref. [50], which can thus be considered an important first step toward the general method that we discuss here.

Another important aspect of our method is that, in contrast to so-called Bayesian quantum feedback approaches [19,22,51], it does not require to simulate the system dynamics during deployment in an experiment, as the controller is only provided with the measurement outcomes. This feature is important both for real-time control at fast time scales and scalability to more complex systems.

We illustrate the power of feedback GRAPE in a series of different tasks, considering different experimental setups relevant for modern quantum computing employing cavity modes [52]. Although our method is general, we focus on feedback sequences with a modular structure, i.e., where building blocks such as unitaries and measurements are combined in discrete time steps. These are useful scenarios, since they can make it easier to interpret the resulting strategies.

In the following, we first present the general method and then we analyze the numerical examples, discuss aspects of the optimization landscape and scalability, and finally present further extensions.

II. FEEDBACK-GRAPE METHOD

We consider a general dissipative quantum system with feedback (for an overview of the scheme, see Fig. 1). We suppose that measurements are performed at times $t_1, t_2, \ldots, t_N$ and that the evolution is controlled—in a manner to be optimized—based on the corresponding measurement outcomes m_j. Specifically, the control parameter F_θ^j (which might be a vector) applied during the time interval $[t_j, t_{j+1}]$ is a function of all previous measurement results $\mathbf{m}_j = (m_1, \ldots, m_{j-1}, m_j)$. Below, we refer to the set of controls $\{F_\theta^j(\mathbf{m}_j)\}$ for all possible measurement outcomes as a feedback strategy or simply strategy [see Fig. 1(c)]. With this notation, we anticipate that the feedback-control functions $F_\theta^j(\mathbf{m}_j)$ are parametrized, depending on trainable parameters θ that will be optimized via gradient ascent (θ is typically a high-dimensional vector). We assume F_θ^j to be differentiable with respect to θ. Ultimately, the value of F_θ^j will be provided by a neural network (NN) or, alternatively, a look-up table: we comment on these different approaches further below but the present considerations are independent of this aspect. In practice, the control vector F_θ^j collects the parameters of a Hamiltonian or, alternatively, a (series of) unitary gate(s). On a minor note, in some scenarios, during the first time interval $[0, t_1]$, one might apply a control F^0 that does not depend on any previous measurement outcomes but can still be optimized.

With this notation in place, the time evolution of the density matrix of the system, for a particular measurement sequence $\mathbf{m} = (m_1, m_2, \ldots)$, can be written in the general form

$$\hat{\rho}(T|\mathbf{m}) = \Phi^{(\mathbf{m})}(F_\theta^N, F_\theta^{N-1}, \ldots, F_\theta^0)[\hat{\rho}(0)], \qquad (1)$$

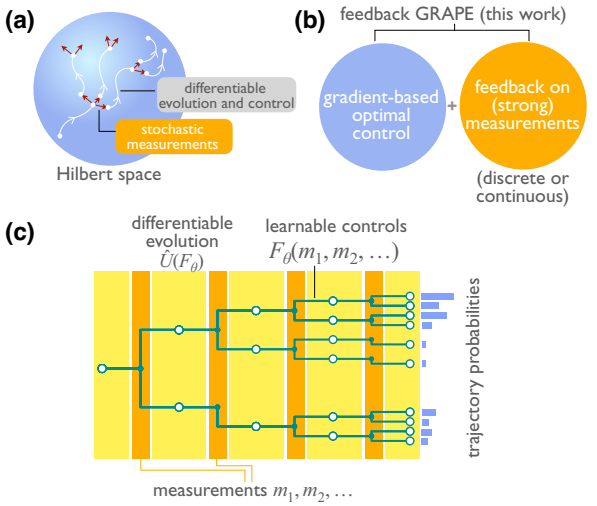

FIG. 1. Quantum feedback control with strong measurements. (a) Such feedback tasks combine smooth differentiable dynamics in Hilbert space with measurement-induced jumps. (b) This work integrates model-based techniques relying on gradients and feedback to strong stochastic measurements. (c) A schematic of the decision-tree representation of a feedback strategy for discrete measurement outcomes. Intervals of differentiable evolution with optimizable control functions F_θ^j depend on the sequence of outcomes $m_1, \ldots, m_j$. In general, the evolution can be dissipative.

where $\Phi^{(\mathbf{m})}$ is the map that depends on the control parameters and implements the quantum dissipative time evolution throughout the whole time interval $[0, T]$, *conditioned on the given fixed sequence* $\mathbf{m}$ *of measurement outcomes*. Note that our definition implies that $\Phi^{(\mathbf{m})}$ itself is not a completely positive (CP) map, because it contains the renormalization of the quantum state required after each measurement (it implements a "quantum instrument"), which introduces a nonlinear dependence on the initial state. To obtain the unconditional average quantum state, the average $\langle \ldots \rangle_{\mathbf{m}}$ of this expression may be taken over all possible measurement sequences, weighted with their respective probabilities.

Equation (1) is valid formally even if the overall evolution is non-Markovian. It can be simplified in the important Markovian case. Then, the evolution proceeds stepwise. Let us denote by Φ_j the CP map for the continuous evolution during the time interval $[t_j^+, t_{j+1}^-]$, where t^- is shorthand for a time point just prior to the measurement at t and, correspondingly, t^+ is right after the measurement. We then have $\hat{\rho}(t_{j+1}^-) = \Phi_j(F_\theta^j(\mathbf{m}_j))[\hat{\rho}(t_j^+)]$. In the special case of unitary dynamics, the evolution itself simplifies further to $\hat{\rho}(t_{j+1}^-) = \hat{U}_j(F_\theta^j(\mathbf{m}_j))\hat{\rho}(t_j^+)\hat{U}_j(F_\theta^j(\mathbf{m}_j))^\dagger$. Here, $\hat{\rho}(t_{j+1}^-)$ is understood to be the quantum state at time t_{j+1} for a fixed sequence $m_1, \ldots, m_j$ of previous measurement outcomes, just prior to the next positive operator-valued measure (POVM) measurement implemented at t_{j+1}.

This measurement is described by some POVM element that can be written in the form $\hat{M}(m')^\dagger \hat{M}(m')$, with the POVM normalization condition $\sum_{m'} \hat{M}(m')^\dagger \hat{M}(m') = 1$ and $\hat{M} \equiv \hat{M}_{j+1}$ depending on the physics of the measurement. It will yield a particular outcome $m_{j+1} \equiv m'$ with probability $P(m') = \mathrm{tr}[\hat{M}(m')^\dagger \hat{M}(m')\hat{\rho}(t_{j+1}^-)]$ and an updated state $\hat{\rho}(t_{j+1}^+) = \hat{M}(m')\hat{\rho}(t_{j+1}^-)\hat{M}(m')^\dagger / P(m')$.

Our goal is to maximize some overall cumulative reward R, which is called the "return" in the nomenclature of reinforcement learning. For example, in a state-preparation task, this might be the final fidelity with respect to some target state $\hat{\sigma}$. For a given sequence $\mathbf{m}$ of outcomes, we would define $R(\mathbf{m}) = \left(\mathrm{tr}\sqrt{\sqrt{\hat{\sigma}}\hat{\rho}(T|\mathbf{m})\sqrt{\hat{\sigma}}}\right)^2$. This would be averaged eventually over all possible measurement outcome sequences to yield $\bar{R} = \langle R(\mathbf{m}) \rangle_{\mathbf{m}}$. The return R could also involve penalties for suppressing larger control amplitudes etc. These additional contributions depend on the specific sequence $\mathbf{m}$ as well, via the controls $F_\theta^j(\mathbf{m}_j)$.

It may now seem straightforward to employ automatic differentiation for optimizing $\bar{R}$ via gradient ascent, updating $\delta\theta = \eta \frac{\partial \bar{R}}{\partial \theta}$, with some learning rate η and with all the trainable parameters combined in a vector θ.

The crucial observation to be made at this stage is that the introduction of stochastic measurement results into this scheme requires some extra care. The following considerations constitute the main conceptual steps needed to enable the discovery of feedback-based quantum control strategies based on gradient ascent.

We have to distinguish between discrete and continuous measurement outcomes, which require substantially different treatment.

For the particularly interesting discrete case (e.g., strong projective qubit measurements), the essential insight is that the probabilities P for obtaining the different measurement outcomes themselves depend on all the controls F_θ^j applied during previous time intervals, simply because the quantum state itself carries this dependence. This has to be taken care of during the evaluation of gradients with respect to θ. Illustrating this in the case of a single measurement m at time $t_1 \in [0, T]$, we have

$$\langle R \rangle_m = \sum_m P(m|\hat{\rho}(t_1^-))R(m). \tag{2}$$

Here, $P(m|\hat{\rho})$ is the probability of measurement outcome m given state $\hat{\rho}$. As we take the gradient with respect to the trainable parameters θ, we observe that the derivative acts not only on the return R based on the time-evolved state $\Phi_1(F_\theta^1(m))[\hat{\rho}(t_1^+)]$ (the second factor inside the sum) but also on the probability $P(m)$ itself, due to its dependence on the initial control, $\hat{\rho}(t_1^-) = \Phi_0(F_\theta^0)[\hat{\rho}(0)]$.

Generalizing this observation, we cannot simply implement gradients of the measurement-averaged return $\bar{R} = \langle R(\mathbf{m})\rangle_{\mathbf{m}}$ by averaging the gradient of the sequence-specific return, $\langle \partial R(\mathbf{m})/\partial\theta\rangle_{\mathbf{m}}$. Rather, observe $\langle R(\mathbf{m})\rangle_{\mathbf{m}} = \sum_{\mathbf{m}} P(\mathbf{m})R(\mathbf{m})$. Thus, when evaluating $\partial\langle R(\mathbf{m})\rangle_{\mathbf{m}}/\partial\theta$, we get two contributions: $\partial[R(\mathbf{m})P(\mathbf{m})]/\partial\theta = P(\mathbf{m})\partial R(\mathbf{m})/\partial\theta + R(\mathbf{m})\partial P(m)/\partial\theta$. To enable stochastic sampling of the second term, we rewrite it using $\partial P(\mathbf{m})/\partial\theta = P(\mathbf{m})\partial \ln P(\mathbf{m})/\partial\theta$. This then leads to

$$\frac{\partial \langle R(\mathbf{m})\rangle_{\mathbf{m}}}{\partial\theta} = \left\langle \frac{\partial R(\mathbf{m})}{\partial\theta}\right\rangle_{\mathbf{m}} + \left\langle R(\mathbf{m})\frac{\partial \ln P_\theta(\mathbf{m})}{\partial\theta}\right\rangle_{\mathbf{m}}. \quad (3)$$

Here, we display explicitly the parameter dependence of $P_\theta(\mathbf{m})$, which represents the probability of the full sequence of outcomes $\mathbf{m} = (m_1, m_2, \ldots)$, given the parameters θ that determined the shape of the control functions F^j_θ.

The mathematics for the extra term appearing here, with the gradient of the log-likelihood, is well known from policy-gradient-based approaches in *model-free* reinforcement learning. However, there this term appears for a different reason. It arises due to the deliberate choice of implementing stochastic controls, in order to avoid any need to take gradients through the possibly unknown dynamics of the system to be controlled (for more details, see Appendix A). In our case, by contrast, we do take gradients through the known dynamics and the controls themselves are deterministic when conditioned on a fixed sequence of measurements. The randomness enters via the stochastic measurement outcomes (these are observations of the "environment" in RL language).

Due to the sequential nature of the control procedure, the log-likelihood term can be rewritten as a sum of contributions, $\ln P_\theta(\mathbf{m}) = \sum_j \ln P_\theta(m_j|\mathbf{m}_{j-1})$. Thus, during the individual time-evolution trajectory, this term may be easily accumulated step by step, since the conditional probabilities are known (these are just the POVM measurement probabilities). The gradients of Eq. (3) can then be taken for such an individual trajectory or a batch, substituting stochastic sampling for an exact average over $\mathbf{m}$. The whole approach, with its calculational pipeline, is illustrated schematically in Fig. 2. Additionally, a more detailed algorithmic flow-chart representation is given in Appendix B.

The Monte Carlo evaluation of the average reward $\langle R(\mathbf{m})\rangle_{\mathbf{m}} = \sum_{\mathbf{m}} P(\mathbf{m})R(\mathbf{m})$ is a crucial ingredient to tackle long sequences of stochastic measurements, as it allows us to focus only on the most likely measurement outcomes among the exponentially large set of such outcomes. Only for very short sequences would it be feasible to instead explicitly evaluate the sum over all possible measurement outcomes, producing less noisy gradients.

The evaluation of the gradients of the return with respect to the trainable parameters θ can proceed in two different ways, using either automatic differentiation (see below) or exploiting analytical approaches to obtain explicit expressions for the gradients that can then be evaluated numerically. In the latter case, one can either set up evolution equations for the parameter gradient of the quantum state, $\partial_\theta\hat{\rho}$ or, in the suitable scenario, directly apply a modified version of the original GRAPE technique to efficiently evaluate the gradients. We describe both of these procedures in detail in Appendix C. In the language of current machine-learning concepts, taking the gradient through the continuous-evolution intervals would be, generally speaking, an example of the concept of neural ordinary differential equations, a rather recent development [53].

Alternatively, and sometimes more conveniently, the whole evolution pipeline described above can straightforwardly be implemented in an automatic differentiation framework, such as TensorFlow [54], PyTorch, JAX, or others. The gradients of the resulting overall return and of the log-probability can then be obtained using that framework without extra effort. The sequence of discrete measurement outcomes of a given trajectory is considered fixed when taking the gradient in this manner. The

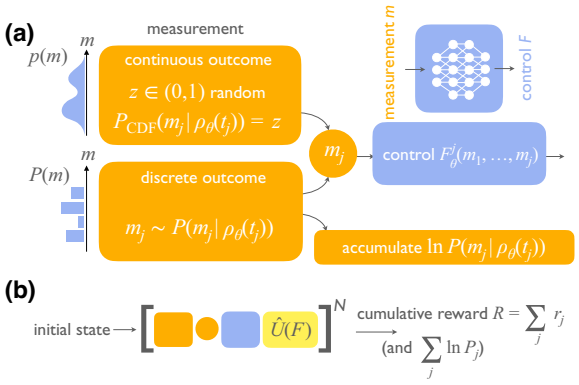

FIG. 2. The quantum feedback sequences considered within feedback GRAPE, set up for automatic differentiation. (a) The measurement samples a stochastic outcome m_t, adopting a different method depending on whether the outcome is continuous or discrete. In both cases, the probability distribution depends in a differentiable way on the trainable parameters θ, via the preceding unitary controls that have generated the present quantum state ρ_θ. Depending on the measurement outcome, a learnable control F is applied that may be implemented either via a neural network (NN) or a look-up table. (b) The full sequence. This consists of repeated application of the blocks depicted in (a), plus subsequent implementation of unitary controls depending on F, potentially with decay and decoherence included in the model of the evolution of the system. For discrete measurement outcomes, the logarithmic term (in brackets) has to be accumulated and is used to evaluate a log-likelihood correction term when optimizing the overall return R using gradient ascent [cf. Eq. (3)]. An algorithmic flow-chart representation of the learning pipeline for the case of discrete measurements is provided in Appendix B.

automatic-differentiation approach is particularly helpful and efficient in cases where the whole time evolution can be split into many building blocks (parametrized gates, i.e., unitaries, acting during fixed time intervals), as is common practice for many quantum control tasks in present quantum computing platforms. Whenever this latter situation is encountered, it also aids interpretability, as we see later, in the numerical examples.

The measurement-dependent controls $F_\theta^j(\mathbf{m}_j)$ are the central quantities of our approach. For accessing those, one can simply adopt a look-up table, at least for the case of discrete measurements discussed up to now and when the total number of measurements during the full time evolution is not too large. The table for F_θ^j needs M^j entries, if there are M possible outcomes for each measurement, corresponding to the exponentially many possible sequences. In that case, the entries of this table would directly represent the trainable parameters θ. Alternatively, the controls $F_\theta^j(\mathbf{m}_j)$ can be implemented via an NN that takes the measurement results as input and maps those to the current control vector. Since the number of available measurement results is different for each time step j, one may choose to set up a different network at each j. However, training efficiency and generalization ability can be improved by constructing a single recurrent network, i.e., a network with memory that is employed in sequence processing tasks [55]. It takes the temporal sequence of measurements as input, one step at a time, producing a control vector at each such time step. This approach can possibly generalize to infinitely long feedback-control sequences, e.g., during state-stabilization tasks. In the course of our numerical experiments, to be detailed later, we observe both scenarios, where the NN outperforms the look-up table but also the reverse.

We note in passing that both the look-up table and the recurrent neural network (RNN) approaches do not require a Bayesian estimate of the state during deployment in an experiment: they operate purely on the measurement outcomes. For the sake of comparison, however, we also consider a Bayesian quantum feedback approach [51] in which a fully connected NN is provided with the quantum state before each time step. This approach suffers problems with scalability, because it can be deployed in an experiment only in combination with real-time simulations of a stochastic master equation [51] to update the quantum state based on the measurement outcomes. We see later that our numerical results indicate that the knowledge of the full quantum state does not appear to provide any substantial learning advantage compared to our non-Bayesian approaches.

Continuous measurement outcomes can be treated in exactly the same way as discrete ones. However, for that scenario there also exists an alternative, which obviates the need for the logarithmic-likelihood correction term: we can adopt a general version of what is known as the "reparametrization trick" in stochastic NNs (e.g., in variational autoencoders). The idea is that we can generate a stochastic variable z according to some *fixed* probability density and then transform this into the required measurement probability density $p(m|\hat{\rho})$, which does depend on control parameters (via the state $\hat{\rho}$, as explained above) and must be subjected to gradients. This parameter-dependent transformation can be implemented in a differentiable way, as we now show. We first obtain the cumulative distribution function $f(m) = \int_{-\infty}^{m} p(m'|\hat{\rho})dm'$, by discretizing p as a vector on a lattice and using a cumulative sum for an Euler approximation of the integral (this operation exists in frameworks such as TensorFlow). We then draw a random uniformly distributed $z \in [0,1]$ and *invert* $f(m)$. The last step also needs to be performed in a differentiable way. One option is to set $m = f^{-1}(z) \approx \sum_n \tilde{m}_n H(z - z_n)H(z_{n+1} - z)$. Here, $z_n = f(m_n)$ defines the lattice version of f, H is the Heaviside step function, the sum ranges over the lattice points, and $\tilde{m}_n$ solves the piecewise linearized approximation of $m = f^{-1}(z)$ associated with the interval n: $\tilde{m}_n = (m_{n+1} - m_n)(z - z_n)/(z_{n+1} - z_n) + m_n$. The set of measure zero where the gradient is undefined can be ignored, as is common practice in using activation functions such as rectified linear units in NNs.

In this way, one can implement, within the automatic differentiation framework, measurements of, e.g., discrete variables with continuous outcomes. A typical case would be a qubit measurement with $m = \sigma + \xi$, where $\sigma = \pm 1$ is the qubit state and ξ some measurement noise of density $q(\xi)$. Formally, $p(m|\hat{\rho}) = \sum_\sigma q(m - \sigma)\rho_{\sigma\sigma}$, and $\hat{M}(m) = \sum_\sigma \sqrt{q(m-\sigma)}\,|\sigma\rangle\langle\sigma|$. One can also perform measurements on continuous variables, e.g., a weak measurement of position, $p(m|\hat{\rho}) = \int dx q(m - x)\rho(x,x)$, with $\hat{M}(m) = \int dx \sqrt{q(m-x)}\,|x\rangle\langle x|$. The dependence of the probability density p in each case on the parameters determining the control functions at earlier times will be correctly taken into account and one can now use the straightforward formula $\partial\bar{R}/\partial\theta = \langle \partial R(\mathbf{m})/\partial\theta \rangle_{\mathbf{m}}$ for stochastic sampling of the gradient. Note that the discrete-outcome case (above) and the continuous-outcome case can also be easily combined in our approach.

Our reparametrization trick allows us to switch from any arbitrary state-dependent probability density $p(m|\hat{\rho})$ to an easy-to-sample fixed probability density, allowing to obviate the need for the log-likelihood term that is required in the case of discrete outcomes discussed earlier. There is a special limiting case in which an even simpler linear reparametrization achieves the same goal. We are referring to the case in which $p(m|\hat{\rho})$ is a Gaussian that depends on the state of the system only via its mean value $\bar{m}$, $p(m|\hat{\rho}) = \mathcal{N}(\bar{m}(\hat{\sigma}), 1)$. In this case, one can switch to a fixed (easy-to-sample) Gaussian probability density with the reparametrization $z = m - \bar{m}(\hat{\sigma}) \sim \mathcal{N}(0,1)$. This

reparametrization has the advantage that it can be trivially inverted. This kind of description automatically arises in the well-known quantum trajectories approach applied to homodyne detection [49] of light emerging from a cavity. In that setting, the observable m of interest is the observed homodyne detection current, appropriately rescaled and integrated over a time window Δt much smaller than the typical system decay time. Its distribution depends on the state of the system only via its mean value $\bar{m} = \epsilon \mathrm{tr}(\hat{O}\hat{\rho})$, where $\hat{O}$ is the system observable that couples to the bath and $\epsilon \propto \sqrt{\Delta t}$. In the framework of this quantum trajectory approach, one can thus find feedback strategies conditioned on the homodyne current by differentiating through the system dynamics without introducing our more flexible reparametrization of $p(m|\hat{\rho})$ or our additional log-likelihood term. This approach has been pursued recently by Schäfer *et al.* [50].

So far, controls have been continuous and represented via functions (differentiable with respect to parameters) depending on previous measurement results. However, sometimes one might want to *also* take discrete actions, e.g., deciding whether or not some measurement should be performed at all or whether some fixed qubit gate should be applied. This can be incorporated without any substantial changes to the approach discussed here, borrowing from policy-gradient model-free reinforcement learning, by introducing stochastic actions a, in contrast to the deterministic continuous actions discussed so far: use an NN or a look-up table to calculate the probability $P_\theta(a_j|\mathbf{m}_j, \boldsymbol{a}_{j-1})$ of taking a discrete action a_j at step j given the previous measurement record $\mathbf{m}_j$ and actions $\boldsymbol{a}_{j-1} = (a_{j-1}, \ldots, a_2, a_1)$ and then sample from all actions accordingly. Then, the same form for the gradient for the reward applies as in Eq. (3) but now with both the reward R and the probability P_θ depending not only on the measurement history $\mathbf{m}$ but also on the history of all the actions taken throughout the trajectory, $\boldsymbol{a} = (a_1, a_2, \ldots)$. As before, the probability in Eq. (3) can be replaced by Monte Carlo sampling, while the log-likelihood term can be accumulated according to $\ln P_\theta(\mathbf{m}, \boldsymbol{a}) = \sum_j \ln P_\theta(m_j|\mathbf{m}_{j-1}, \boldsymbol{a}_{j-1}) + \ln P_\theta(a_j|\mathbf{m}_j, \boldsymbol{a}_{j-1})$.

III. NUMERICAL EXAMPLES

We now turn to an illustration of the feedback-GRAPE method by solving several different challenging quantum feedback-control tasks. We consider five separate tasks of increasing difficulty: starting with noiseless state preparation (an open-loop control task) as a baseline benchmark for GRAPE-type control in this scenario, then moving to purification (a task that already benefits from feedback, i.e., adaptive measurements), to feedback-based state preparation in the presence of noisy control parameters or out of a thermal state, and feedback-based state stabilization. Along the way, we explore a handful of different experimental scenarios.

A. State preparation with Jaynes-Cummings controls

As a preliminary step, we consider state preparation of a target state starting from a pure state. In addition, we assume that any coupling to an external environment is negligible and that the parametrized controls can be implemented perfectly. In this setting, the preparation of a quantum state does not require any feedback and, thus, we will not yet be able to test our feedback extension of GRAPE. Instead, the purpose of this section is to provide a compelling motivation for our approach, showing that, even before feedback is introduced, (GRAPE-type) model-based optimal-control approaches outperform, sometimes dramatically, their model-free counterparts.

As a first example, we consider the state preparation of a cavity resonantly coupled to an externally driven qubit [cf. Fig. 3(a)]. This scenario is modeled by the well-known Jaynes-Cummings Hamiltonian [56]. It is the first and simplest light-matter coupling scenario that emerged in quantum optics [56,57] but it is nowadays of practical relevance for modern quantum computing platforms [52]. In those, it is employed both for qubit readout and for qubit-enabled nonlinear manipulation of cavity states. Here, we consider a particular sequence of parametrized unitary gates originally introduced by Law and Eberly [58]. The sequence consists of a series of two interleaved gates [cf. Fig. 3(b)]. In the first gate, the

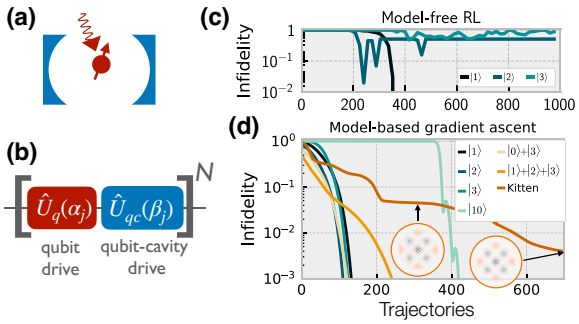

FIG. 3. State preparation from the ground state (open-loop control). (a) A schematic of the Jaynes-Cummings system. (b) The sequence of parametrized controls during a time-evolution trajectory. Each trajectory comprises N time steps. A time step consists of a qubit drive gate followed by a qubit-cavity interaction gate. (c),(d) The fidelity as a function of the number of trajectories sampled during training (here, that number is equal to the number of gradient-ascent iterations). The target state is indicated. (c) Model-free RL performs very poorly, while (d) the direct gradient-based approach used as the basis for our method converges well even when long sequences are required ($N = 20$ for the four-component kitten state).

qubit is driven externally to implement an arbitrary rotation about an equatorial axis, implementing the unitary gate $\hat{U}_q(\alpha) = \exp[-i\left(\alpha\hat{\sigma}_+^c + \alpha^*\hat{\sigma}_-^c\right)/2]$. Here, we introduce the qubit-raising (-lowering) operator $\hat{\sigma}_+^c$ ($\hat{\sigma}_-^c$) and $|\alpha|$ is the rotation angle while $\arg(\alpha/|\alpha|)$ is the azimuthal angle of the rotation axis. In the second gate, the qubit and cavity mode with ladder operator $\hat{a}$ can be coupled for a variable duration, exchanging excitations, $\hat{U}_{qc}(\beta) = \exp[-i\left(\beta\hat{a}\hat{\sigma}_+^c + \beta^*\hat{a}^\dagger\hat{\sigma}_-^c\right)/2]$. Here, $|\beta|$ is proportional to the interaction time (see Appendix D). Depending on the target state, the control parameters α and β can be chosen to be complex or real. In the latter case, the control vector F^j defined in Sec. II is simply $F^j = (\alpha_j, \beta_j)$.

In their groundbreaking work [58], Law and Eberly have shown that any arbitrary superposition of Fock states with maximal excitation number N can be prepared out of the ground state in a sequence of N such interleaved gates, also providing an algorithm to find the correct angles and interaction durations (see Appendix D). This solution has been used to remarkable effect in experiments with superconducting qubits [59]. Here, we use it as a benchmark to test different RL approaches.

With the goal of recovering the strategies predicted by Law and Eberly, we set the return R equal to the state fidelity at the final time step, prescribing a fixed number of time steps equal to the maximum number of excitations in the target state, e.g., for the state $\propto |1\rangle + |3\rangle$, we set $N = 3$.

Somewhat surprisingly, state-of-the-art model-free reinforcement learning is unable to cope well with this challenge. We employ proximal-policy optimization (PPO) [60], a powerful and widely used modern general-purpose advantage actor-critic approach to optimize the continuous controls. It performs well only for the very simple task of preparing Fock state $|1\rangle$, while getting stuck at bad final overlaps for higher Fock states [cf. Fig. 3(c)]. This statement holds even after training for many episodes, varying the hyperparameters, and even applying other modern model-free RL algorithms (see Appendix E).

In contrast, direct gradient ascent through the unitary evolution, using the control parameters as learning parameters, $\theta = \{F^j = (\alpha_j, \beta_j)\}$, allows us to find optimal state-preparation strategies performing as well as the known Law-Eberly algorithm for a broad range of states. As examples, we prepare Fock states with excitation numbers up to $n = 10$ and superpositions of two Fock states [cf. Fig. 3(d)]. For all these states, convergence of the training protocol is obtained in a single run and the infidelity can be decreased up to the numerical precision of the algorithm.

In addition, we also consider a much more challenging four-component kitten state built from four coherent states, $|\psi_{\tilde{\alpha}}^{\mathrm{Kit4}}\rangle \propto \sum_{j=0}^{3} |i^j\tilde{\alpha}\rangle$. Here, we consider $\tilde{\alpha} = 3$, corresponding to the average photon number $\bar{n} \approx |\tilde{\alpha}|^2 = 9$. We attempt to prepare this state using a long sequence of 20 time steps. In this case, we see that the infidelity tends to decrease stepwise during the training [cf. Fig. 3(d)]. In Appendix F, we show that the detailed evolution of the fidelity during training depends strongly on the initialization. On the other hand, the height of the steps is an intrinsic feature of the target state. Interestingly, each step can be associated with a particular intermediate state that can be reached using a large number of different strategies. Each such strategy corresponds to a saddle point of the optimization landscape. The training becomes particularly slow close to these saddle points because the curvature of the optimization landscape in the direction of increasing fidelity is zero, giving rise to a narrow valley. This can cause the training to stall on a suboptimal solution. Nevertheless, if the preparation sequences are made longer (larger N), good solutions can be found in any training run (for more details, see Appendix F).

Physically, the presence of a large number of narrow valleys and plateaus in the optimization landscape is due to the fact that in the Law and Eberly protocol, the excitations can only be added one by one, first exciting the qubit and then transferring them to the oscillator. They are present even for simple tasks such as the preparation of a simple Fock state with excitation number n prepared in $N = n$ time steps [as in Figs. 3(c) and 3(d)]. In this case, it is enough that a single control parameter (β_j or α_j for any j) is zero for the fidelity and its gradient to vanish. If two or more parameters are small, the Hessian is also zero, leading to a plateau. In the model-based approach, this leads to a slow convergence for large n [cf. Fig. 3(d)]. On the other hand, lack of direct access to the gradient in a model-free approach prevents convergence even for small values of n [cf. Fig. 3(d)].

In order to further substantiate that model-based GRAPE-type optimal control approaches are more efficient than their model-free counterparts, we analyze the state preparation of a so-called Gottesman-Kitaev-Preskill state [61], using the same set of universal controls as recently adopted by Sivak *et al.* [38] to demonstrate their model-free quantum control approach. Direct comparison of our results, reported in Appendix G, with those of Ref. [38] shows the following: in this setting with more powerful controls, in which model-free RL performs already well, a GRAPE-type approach performs even better. In particular, it allows us to explore much larger parameter spaces and, thus, obtain better-quality solutions using only a small fraction (about 1%) of simulated training trajectories.

Regardless of these detailed observations, these examples indicate that model-based gradient-ascent approaches can outperform model-free generic methods for optimizing quantum control in settings relevant for quantum technologies. Given the large performance difference in the open-loop control scenario already, we focus entirely on

the feedback-GRAPE approach in the subsequent exploration of the more advanced challenges that do include feedback.

B. State purification with qubit-mediated measurement

Next, we move to a first example of a situation that requires feedback. We now imagine that the cavity is initially in a mixed state. The goal is to purify the state of the cavity, i.e., the reward is determined by the purity $\mathrm{tr}\hat{\rho}_{\mathrm{cav}}^2$ of the cavity state at the final time. Purification can be achieved by applying repeated quantum measurements, which remove entropy from the quantum system.

In the following, we consider an adaptive measurement scheme originally proposed in Ref. [62] and demonstrated in a series of experiments on Rydberg atoms interacting with microwave cavities [21,63–65]. In this scheme, the cavity is coupled to an ancilla qubit, which can then be read out to update our knowledge of the quantum state of the cavity [cf. the sketch in Fig. 4(a)].

The measurement comprises several steps, which we list individually before summarizing their combined effect on the cavity state. In a first step, the ancilla qubit with Pauli operators $\hat{\sigma}^a_{i=x,y,z}$ is prepared in the $+x$ eigenstate. Subsequently, it is coupled dispersively to the cavity for a variable amount of time. The dispersive coupling in experiments is linear in the photon number to a very good approximation in the low-photon regime and is described by a unitary of the form $\hat{U}(\gamma) = \exp(-i\gamma\hat{\sigma}^a_z\hat{a}^\dagger\hat{a})$, with parameter γ depending linearly on the interaction time. This means that the qubit precesses by an angle that depends linearly on the number of photons inside the cavity. In the next and final step, the ancilla qubit is projected along some selected axis $\hat{\sigma}^a_x\cos\delta + \hat{\sigma}^a_y\sin\delta$, yielding a discrete result $m \in \{-1,+1\}$. The combined effect of these operations is to perform a POVM on the cavity, with outcome probability $P(m) = \mathrm{tr}[\hat{M}(m)^\dagger\hat{M}(m)\hat{\rho}]$ and an updated state $\hat{M}(m)\hat{\rho}\hat{M}(m)^\dagger/P(m)$. Here, $\hat{\rho}$ is the state of the cavity, excluding the measurement qubit, which is eliminated in this description. The measurement operator $\hat{M}(m)$ is given by

$$\hat{M}(m=+1) = \cos(\gamma\hat{a}^\dagger\hat{a} + \delta/2), \tag{4}$$

and likewise for $m=-1$, with cos replaced by sin. This formula indicates that after the measurement, the probabilities of the different cavity Fock states $|n\rangle$ are multiplied by a sinusoidal "mask," where the period is determined by $1/\gamma$ and the phase shift is set by both δ and the measurement outcome m. This helps to pinpoint the state of the cavity, especially when multiple such measurements are carried out with suitably chosen periodicities [62] and phase shifts [65].

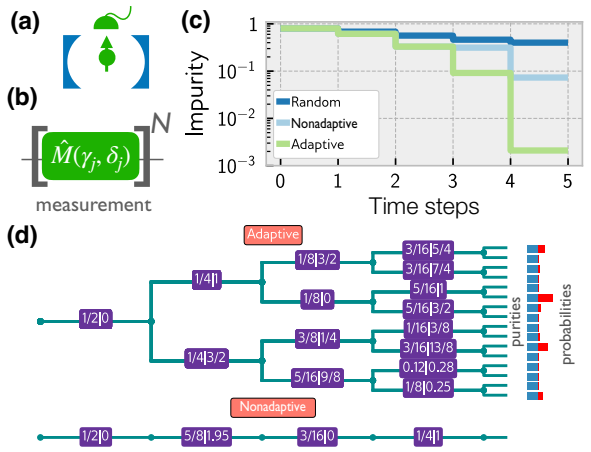

FIG. 4. Purification of a thermal state. (a) A sketch of the experimental setup. The cavity is coupled dispersively to an ancilla qubit. (b) The sequence of parametrized controls during a time-evolution trajectory. Each time step consists of a parametrized measurement. (c) Purity as a function of time starting from a thermal state with $\bar{n}=2$. An adaptive strategy (found using feedback GRAPE) clearly outperforms other approaches. (e) The extracted purification strategy visualized in a decision tree. The purple boxes display the measurement parameters $(\gamma/\pi|\delta/\pi)$.

Figure 4(c) shows the results of applying the feedback-GRAPE method to this problem (labeled "Adaptive"). We employ an RNN to produce the controls $F^j = (\gamma_j, \delta_j)$ when provided with the measurement outcome sequence (more details on numerical parameters can be found in Appendixes D and E). As we see, the impurity quickly decreases with the number of allowed measurements and it does so significantly better than in a nonadaptive scheme, where the sequence of measurement controls δ_j and γ_j is still optimized but where these controls are not allowed to depend on previous measurement outcomes. To visualize and analyze the numerically obtained strategy, in Fig. 4(d) we introduce a decision tree. This is extracted via an automated numerical procedure, by running many trajectories and noting in each case the controls suggested by the adaptive strategy. The controls are a deterministic function of previous measurement outcomes.

Such a decision tree will contain all information about the adaptive strategy learned by the NN and can possibly allow the user to give it a physical interpretation and extrapolate analytical solutions for larger numbers of control steps. This may require us to leverage any available physical understanding of the control operations, e.g., identifying physically significant values of the control parameters. Using our understanding of the physics of the model, we can choose to (analytically) interpret the controls, e.g., try to represent them in terms of fractional multiples of π. This kind of analysis is optional and independent of our method but it nicely demonstrates what can usefully be done in settings with discrete measurements,

generating additional insights after running the general-purpose algorithm. For example, here, we are able to take inspiration from the decision tree for four measurements and a specific value of the temperature to extrapolate the optimal purification strategy for any temperature and any number of measurements (see Appendix J).

C. State preparation from a thermal state with Jaynes-Cummings controls

We now turn to a task that involves both feedback and control simultaneously. Specifically, we consider state preparation out of a thermal state, for target states that are selected as arbitrary superpositions of the first few Fock states. For this purpose, we consider the setup shown in Fig. 5(a), comprising both an ancilla and a control qubit to combine the parametrized measurements introduced in Sec. III B with the Jaynes-Cummings control gates introduced in Sec. III A. The resulting sequence of parametrized controls is shown in Fig. 5(b).

Results for the state $(|1\rangle + |2\rangle + |3\rangle)/\sqrt{3}$ and a four-component kitten state $|\psi_{\tilde{\alpha}}^{\mathrm{Kit4}}\rangle$ with $\tilde{\alpha} = \sqrt{2}$, are shown in Figs. 5(c) and 5(d). Feedback GRAPE converges in about 1000 gradient-ascent steps (each operating on a batch of ten sampled trajectories). We run the method several times, starting with different initial random configurations of the trainable parameters θ, demonstrating that convergence is robust, despite the usual absence of a guarantee for such a nonconvex optimization problem (for more details, see Appendix I).

It is interesting to analyze the convergence behavior in more detail. As one noteworthy observation, despite the overall very good performance, we sometimes find that the algorithm may get stuck at suboptimal solutions if we increase the total number of time steps available for the feedback sequence [Fig. 5(d)]. Ideally, an increased number of steps should always lead to an improvement (in the present scenario) but apparently the larger space of control variables then becomes challenging. This can be mitigated to some extent by running the gradient ascent repeatedly from random starting conditions. We discuss other possible solutions to this general problem in Sec. IV.

One motivation for the use of an NN instead of a look-up table is that the number of parameters needed for a tree-type table grows exponentially, while an NN could in principle make use of a much smaller number of parameters. Also, it may be expected that the strategy of a network generalizes to situations with a number of time steps larger than the one on which it was trained. Despite these obvious advantages of NNs, we find (to our surprise)

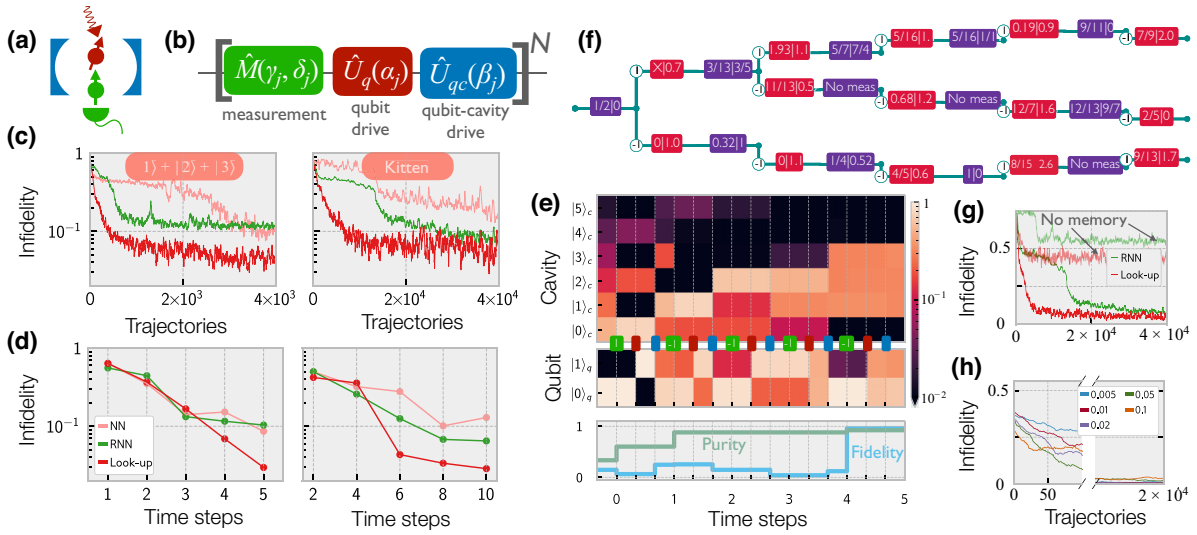

FIG. 5. State preparation from a thermal state (with average occupation $\bar{n} = 1$), employing feedback. (a) A schematic of the Jaynes-Cummings system with an additional ancilla qubit used for the measurement. (b) The sequence of parametrized controls. Each trajectory corresponds to N time steps. Each time step consists of a parametrized measurement, followed by two unitary gates. (c) The gradient-ascent progress for two target states (the curves are smoothed with a moving average). (d) The final infidelity versus the total number of time steps. In (d), each point is the best out of 30 training runs. The statistics of the final infidelity for random training initialization are analyzed in Appendix I. (e) The evolution of the reduced qubit and cavity state (probability as color) for one trajectory of the converged strategy (target $|1\rangle + |2\rangle + |3\rangle$); time points of measurements (with results) and controls are indicated as in (b). (f) The corresponding decision tree, for the most probable sequences of measurement outcomes. The red boxes show $(\alpha/\pi\,|\,\beta/\pi)$ and "No meas" means that the parameters γ and δ are such that no measurement takes place. Gradient-ascent progress [for the same state and number of steps as in (e) and (f)] (g) with and without memory and (h) for different values of the learning rate (see legend).

that look-up tables often converge to better fidelities than RNNs in the present example scenario with feedback [cf. the red and green lines in Figs. 5(c) and 5(d)]. Another important observation is that both the look-up table and the RNN methods fare at least equally well if not better than a quantum state aware NN controller (the pink line in Figs. 5(c) and 5(d). As mentioned above, these methods are preferable as they do not require any real-time simulations during deployment in an experiment. The reasons for these observations are still unclear and merit future investigation.

What is the nature of the feedback strategies that the algorithm discovers? Naively, we might expect the following strategy: an optimized adaptive purification phase, of the kind discussed above, leading to some Fock state $|n\rangle$, followed by state preparation that is derived from the Law-Eberly protocol (e.g., going back down to the ground state and then building up the arbitrary target state from there). However, the actual strategies discovered by feedback GRAPE are significantly more efficient. They already interleave adaptive measurements and controls in the first stage of the process. This can be seen in Figs. 5(e) and 5(f), where the goal is to prepare the equal superposition $(|1\rangle + |2\rangle + |3\rangle)/\sqrt{3}$. Again, it is possible to obtain more information about the full strategy (as opposed to a single trajectory), by extracting a decision tree [Fig. 5(f)]. There, we observe that measurements are sometimes deliberately performed in such a way that certain Fock states are completely ruled out (their probability is set to 0), which requires certain choices of measurement control parameters. Simultaneously, qubit-cavity interaction cycles are employed to reduce the excitation number of the cavity.

D. State stabilization in a noisy environment with Jaynes-Cummings controls

Quantum state stabilization in a noisy environment represents another challenging task that can be solved using feedback GRAPE.

In this scenario, the interaction with the environment induces decay and decoherence of the quantum state. Both these effects can be suppressed by probing the system with an appropriate stream of quantum measurements interleaved with unitary gates, leading to the long-term stabilization of the quantum state.

We use a similar Jaynes-Cummings feedback-control scheme as in Sec. III C, here, allowing for multiple control substeps. Each substep comprises a qubit drive and a qubit-oscillator interaction gate. In addition, we model the physical decay of the cavity with decay rate κ, interleaving the substeps of the feedback-control sequence with intervals of dissipative evolution of fixed durations t_M and t_c [cf. Fig. 6(a)]. These could be interpreted as waiting times before applying instantaneous measurement and control gates, respectively, but more realistically they can

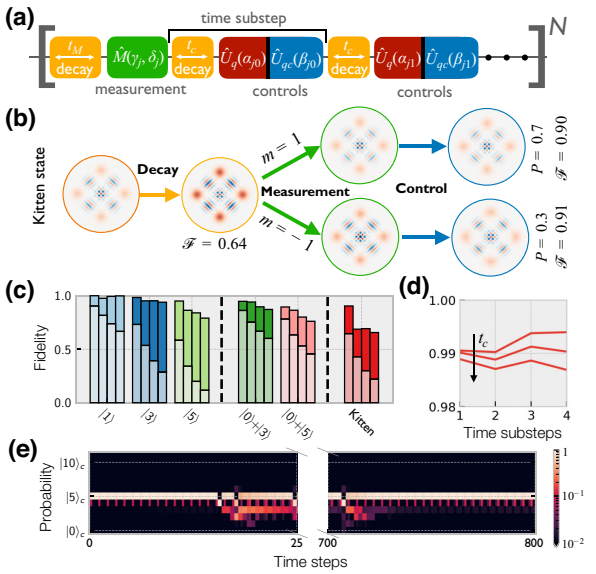

FIG. 6. State stabilization with Jaynes-Cummings controls. (a) A sketch of the feedback-control sequence, also including the physical decay of the cavity. We assume that multiple control substeps can be applied after each measurement. Each substep comprises a qubit driving gate followed by a qubit-cavity interaction gate. The decay with rate κ is incorporated by interleaving the parametrized controls with dissipative evolution of fixed durations t_M and t_c before a measurement and control substep, respectively (for more details, see Appendix H). (b) Trajectories after optimization. We show the evolution of the oscillator Wigner function during one step of the feedback-control sequence stabilizing a four-legged kitten state with average photon number $\bar{n} = 9$. Also indicated are the probabilities P of each measurement outcome as well as the fidelity $\mathcal{F}$ after the decay and control steps. (c) The performance of the strategy found by feedback GRAPE. We show the final fidelity for various target states and numbers of steps N. After each decay and measurement, a single control sequence (i.e. only one choice of α and β) is applied. The four columns represent different numbers of decay steps experienced by the state (here, $N = 1, 2, 3$ and 4). The bars with lower values show the bare decay of the fidelity (here, for $\kappa t_M = 0.05$ and $t_c = 0$), when no feedback strategy is employed. (d) For sufficiently low dissipation (small t_c), the fidelity can be increased by applying more control substeps. We show the fidelity as a function of the number of substeps for $N = 1$, $\kappa t_M = 0.1$ and three different decay durations t_c, $\kappa t_c = 0$, 1×10^{-4}, and 2×10^{-4}. (e) The stabilization of a Fock state (here, $|5\rangle$) for an arbitrarily long time, employing the generalization ability of a recurrent neural network (RNN).

effectively incorporate the decay and decoherence that has occurred during finite-duration operations. For this approximation to work, the decay needs to be weak, which is the case for our scenario.

As an illustrative example, we show the two possible trajectories for a circuit comprising a single step of the feedback-control sequence and optimized to stabilize a

four-legged kitten state $|\psi_{\tilde{\alpha}}^{\mathrm{Kit4}}\rangle$ with average photon number $\bar{n} \approx 9$ [cf. Fig. 6(b)]. The learning algorithm selects the measurement parameters $\delta = 0$ and $\gamma = \pi/2$. This seems a natural choice because in this case, $\hat{M}(m = 1)|\psi_{\tilde{\alpha}}^{\mathrm{Kit4}}\rangle = |\psi_{\tilde{\alpha}}^{\mathrm{Kit4}}\rangle$. This implies that the measurement leaves the target kitten state invariant. In addition, a measurement outcome $m = -1$ postselects an orthogonal state. The latter outcome is verified if, e.g., a single-photon leaks out of the cavity and occurs with 30% probability after the dissipative evolution of duration $\kappa t_M = 0.05$. The fidelity recovers up to 91% after a single Jaynes-Cummings control sequence is applied. This is a good result considering the limited expressivity of our control scheme. The fidelity can be moderately increased by allowing more steps provided that the decay during the control protocol is not too large [cf. Fig. 6(d)].

The fidelity for several quantum states, including Fock states and superpositions thereof, for a varying number of time steps (up to four) is shown in Fig. 6(c). As a comparison, the bare decay in the absence of any control is also displayed. These results demonstrate the ability of feedback GRAPE to discover strategies to mitigate the effect of dissipation for a variety of quantum states.

As we explained above, look-up tables often perform surprisingly well. We now briefly demonstrate, in the context of state stabilization, one example where the power of an NN is clearly helpful [Fig. 6(e)]. We first train an RNN on sequences of 20 steps, with the goal of stabilizing a given Fock state for an arbitrarily long time. For this example, the cumulative reward of a trajectory is not only the final fidelity but the sum of fidelities at all time steps. After training, we test on a 40-times-longer simulation and we see that the strategy learned by the RNN generalizes well even for longer sequences. We note how the strategy can recover, even when some "unlucky" measurement outcomes significantly perturb the quantum state.

E. State stabilization with SNAP gates and displacement gates

The use of feedback GRAPE applied to the Jaynes-Cummings scenario allows us to discover strategies extending the lifetime of a range of quantum states. However, for more complex quantum states such as kitten states, the infidelity becomes significant after just a few dissipative evolution steps in spite of the feedback [cf. Fig. 6(c)]. This raises the question of whether the limited quality of the stabilization is to be attributed to a failure of our feedback-GRAPE learning algorithm to properly explore the control-parameter landscape or, rather, to the limited expressivity of the controls. With the goal of addressing this question, we test our method on the state-stabilization task using a more expressive control scheme. Specifically, we use a universal control scheme originally proposed in Ref. [66]. This consists of a sequence of interleaved *selective number-dependent arbitrary-phase* (SNAP) gates $\hat{S}(\{\varphi_n\})) = \sum_n e^{i\varphi_n}|n\rangle\langle n|$ and displacement gates $\hat{D}(\alpha) = \exp[\alpha\hat{a}^\dagger - \alpha^*\hat{a}]$. This is the same control scheme as adopted by Sivak *et al.* [38] to demonstrate their model-free optimal-control approach for state preparation and, as mentioned above, we also use it to demonstrate the preparation of a so-called Gottesman-Kitaev-Preskill state [61] with open-loop controls (see Appendix G). We now go one step further and employ these powerful controls inside a feedback-based state-stabilization scheme, optimized via feedback GRAPE.

As a test example, we consider the feedback-based stabilization of a kitten state built from two coherent states, $|\psi_{\tilde{\alpha}}^{\mathrm{Kit2}}\rangle \propto |\tilde{\alpha}\rangle + |-\tilde{\alpha}\rangle$ with $\tilde{\alpha} = 2$, corresponding to an average photon number of $\bar{n} \approx |\tilde{\alpha}|^2 = 4$. This state has even parity $\hat{P}|\psi_{\tilde{\alpha}}^{\mathrm{Kit2}}\rangle = |\psi_{\tilde{\alpha}}^{\mathrm{Kit2}}\rangle$ with $\hat{P} = \exp[i\pi\hat{a}^\dagger\hat{a}]$. After an excitation leaks out of the cavity, it decays into an odd cat state with the same $\tilde{\alpha}$ [cf. Fig. 7(b)]. Thus, we can detect these decay processes using repeated parity measurements. After such a process, an optimized control step can transform the odd kitten state back into the target kitten state with high fidelity [cf. Fig. 7(b)]. These considerations motivate us to use the feedback-control sequence shown in Fig. 7(a). We use the real and imaginary parts of the phase-space displacements α^j as control parameters, together with the phases φ_n^j for the first N_{SNAP} Fock states (the remaining phases are set to zero).

We train several NNs using as a reward the time-averaged fidelity, $R = \sum_j^N \mathcal{F}_j/N$ with $\mathcal{F}_j$ calculated after applying the block of unitary gates. We consider two different durations N_t of the quantum trajectories seen during training. We also consider two different scenarios for our description of the dissipative evolution. In a first scenario, the dissipative evolution is concentrated before the measurement, $t_c = 0$ and $\kappa t_M = 0.01$. In a second scenario, the dissipative evolution is subdivided into two intervals before and after the measurement, $\kappa t_M = \kappa t_c = 0.005$. Finally, we test the performance of the NNs in stabilizing the kitten state for $N_i = 200$ time steps. This is much larger than the number of time steps seen during training, $N_t = 2$ or $N_t = 10$, and the typical decoherence time t_d for our kitten state, $\kappa t_d = \bar{n}^{-1} \approx 1/4$ corresponding to approximately 25 time steps [cf. Figs. 7(c) and 7(d)].

Our results obtained using the best-performing NNs for each of the four scenarios discussed above are summarized in Figs. 7(c) and 7(d). Figure 7(d) shows the infidelity as a function of time, averaged over a representative set of trajectories. The infidelity is plotted as a solid line in the time interval seen during training (and dashed thereafter). As a comparison, the infidelity without any stabilization (gray line) and the infidelity after a single interval of dissipative evolution of duration $(t_c + t_M)/2$ (the lower edge

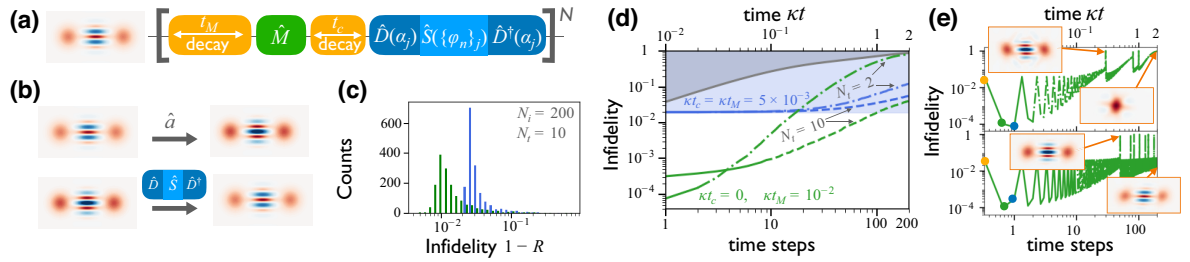

FIG. 7. State stabilization with more expressive controls. (a) A sketch of the feedback-control sequence. A time-evolution step consists of a parity measurement, interleaved between two dissipative evolution substeps and a block of unitary gates comprising two phase-space displacement gates interleaved with a SNAP gate. (b) The evolution for an even kitten state. The decay of a single excitation induces the transition to an odd kitten state. An optimized feedback sequence can transform this state back into the original kitten state with high fidelity. (c) Histograms showing the probability distribution for the time-averaged infidelities of single quantum trajectories for $N_i = 200$ time steps during inference, while training runs only have $N_t = 10$ time steps. The histogram colors identify two different dissipation scenarios. The durations t_c and t_m of the dissipative evolution substeps are displayed in (d). The underlying feedback strategies are predicted by two different NNs, chosen as the best performing out of five training runs. (d) The infidelities as a function of time, averaged over a representative set of trajectories. We consider two dissipation scenarios (color) and two values of N_t. The curves are plotted as solid lines in the time interval seen during training. The NNs used to predict the feedback strategies are the best performing out of five training runs. Also shown is the infidelity in the absence of any control (gray) and after dissipative evolution for a duration $(t_c + t_M)/2$ (edge of blue-shaded region) (e) The time evolution of the infidelity for two typical inference trajectories, predicted using NNs trained on trajectories of duration $N_t = 2$ (top) and $N_t = 10$ (bottom). Here, we also show the infidelity after each substep. The first dissipative (measurement) substep is marked as a yellow (green) dot. Also shown are the Wigner functions after the first odd-parity measurement and at the end of the time evolution.

of the blue-shaded region) are also shown. The latter represents a theoretical lower bound for the infidelity in the scenario with $t_c = t_d$ (as the effects of dissipation after the last measurement cannot be corrected). Figure 7(c) shows a histogram for the distribution of the time-averaged infidelity $1 - R$ for single quantum trajectories.

From these results, we can generally conclude that the feedback strategies discovered using feedback GRAPE allow us to maintain a low infidelity for measurement sequences, much longer than those seen during training, demonstrating a remarkable power of generalization. For the scenario with dissipation injected after the measurement (blue lines), the fidelity remains just above the theoretical lower bound for much of the time evolution. In both dissipation scenarios, the NNs trained on longer measurement sequences perform better. This tendency is most evident in the dissipative scenario with $t_c = 0$. In this case, the NN trained on sequences of just two measurements performs very well on a similar time horizon. However, its ability to generalize the feedback strategy to a longer time evolution is poorer.

In order to better understand this behavior, we plot the time evolution of the infidelity for two typical trajectories (one for each NN), showing also the infidelity after each substep (decay, measurement, or control) [cf. Fig. 7(e)]. From these results, one can see that the strategies learned by the two NNs are qualitatively different. The NN trained on shorter measurement sequences pursues a greedy strategy that decreases the infidelity after each block of unitary gates (top). In contrast, the NN trained on a longer measurement sequence learns a nongreedy strategy, which increases the infidelity after even-parity measurements (bottom). Another obvious difference between the two strategies is that the latter more often triggers odd-parity measurements, which are imprinted in the infidelity as peaks of unit height [cf. Fig. 7(e)]. This is a good sign because it indicates that the parity measurements are able to extract more of the entropy injected during the dissipative part of the dynamics or, equivalently, to better suppress decoherence. This is also reflected in the clear interference fringes in the Wigner function of the state after 200 time steps, 8 times larger than the typical decoherence time [cf. inset of Fig. 7(e)].

This example highlights the importance of training on long measurement sequences to develop more robust stabilization strategies.

Overall, we conclude that expressive controls such as the well-established SNAP gate allow feedback GRAPE to discover excellent feedback strategies in an efficient manner and that strategies discovered for shorter training sequences generalize nicely to much longer sequences. To the best of our knowledge, the level of state-stabilization performance demonstrated here has not been achieved with any other method, despite this being an area of active research in the context of quantum devices and quantum error correction.

F. State preparation in the presence of model uncertainty

Until now, we have assumed that a model for the stochastic dynamics of our quantum system is known without any uncertainty. However, in practice, the model parameters are known only with a finite precision; they might deviate from theoretical predictions because of disorder and be difficult to measure precisely. Moreover, they might even be subject to slow drifts because of environment-induced changes in the quantum device. For all these reasons, an important direction of research in optimal control focuses on improving model-based methods to better perform in the presence of model uncertainties [13–16]. Our goal in the present section is to explore how feedback GRAPE can contribute to this challenge. At the same time, this also allows us to study the optimization landscape in a manageable example.

There are two fundamentally different approaches to dealing with model uncertainties in a coherent control setting. In the first "data-driven" approach, experimental data are used during training. In the second "fluctuation-model-based" approach, the model parameters are sampled from a probability distribution during training. The resulting parameter fluctuations reflect an imperfect knowledge of the model parameters and the strategy is optimized for being resilient against these fluctuations.

Several extensions of GRAPE have been proposed to incorporate uncertainty in the model parameters following either the "data-driven" approach [13,14,16] or the "fluctuation-model-based" approach [15]. However, we emphasize that none of these extensions include feedback.

Our feedback-GRAPE method can also be extended to account for parameter uncertainties, by using either a "data-driven" or a "fluctuation-model-based" approach. For the "data-driven" approach, one could follow an approach similar to that of d-GRAPE [14] and c-GRAPE [16], modifying the analytical formula for the learning gradient (see Appendix C) to also incorporate operators estimated in quantum tomography experiments. Alternatively, feedback GRAPE could be used in combination with model-free RL, with both methods sharing the same controller in the form of an RNN. In this setting, feedback GRAPE would be used for the initial training of the controller allowing to explore higher-dimensional control-parameter spaces. Subsequently, the controller will be trained on experimental data using a model-free approach to obtain a more accurate feedback strategy.

In the following, we instead demonstrate the "fluctuation-model-based" approach in more detail. Not only is this approach more straightforward to implement but it is also best suited to a scenario with feedback: since the measurement statistics depends on the model parameters, the measurement outcomes carry information about the underlying model parameters. At the same time, the strategies are conditional on the measurement outcomes and, thus, can be adapted to the most likely underlying model parameters. This approach is so powerful that it is sometimes worthwhile adopting it even in a model-free RL approach to forgo costly training on experimental data. A prominent example of this is the control of tokamak plasma, where zero-shot transfer from simulations to hardware with imprecisely known parameters has been demonstrated [67].

We consider a simple toy model in which an inhomogeneous ensemble of qubits initially in the ground state is subject to a series of N pulses interleaved with projective measurements on their computational basis [cf. Fig. 8(a)]. The duration of the pulses, $\{\tau_j\}$, can be controlled but the coupling g of a qubit to the driving field is a random variable, distributed according to a Gaussian distribution of average $\bar{g}$ and standard deviation σ [cf. Fig. 8(b)]. As a consequence, the Bloch-sphere rotation angles $\alpha_j = g\tau_j$ will also be Gaussian random variables, now with standard deviations $\tau_j\sigma$ [cf. Fig. 8(a)].

Our goal is to maximize the number of qubits correctly flipped to their excited state or, equivalently, the fidelity averaged over the measurement outcomes and the coupling g, $\langle\langle\mathcal{F}_N\rangle_{\mathbf{m}\sim P(\mathbf{m}|g)}\rangle_{g\sim P(g)}$. In this case, we are interested in the optimal solution for a fixed number of pulses N.

Before discussing this problem in general, let us consider the limiting case of only one time step, $N = 1$. In this case, the pulse duration τ_0 is the only control parameter and there is no feedback by any previous measurement. The average fidelity as a function of this parameter is shown in Fig. 8(c). In the limiting case without fluctuations, a single π pulse of duration $\tau_0 = \pi/g$ or any of its odd integer multiples will achieve zero infidelity [cf. the dashed line in Fig. 8(c)]. Once parameter fluctuations are introduced, one might still expect $\tau_0 = \pi/\bar{g}$ and any of its odd-integer multiples to be optimal. In this way, the spins with coupling $g = \bar{g}$, corresponding to the peak of the parameter distribution $P(g)$, would be flipped with unit probability. However, we observe that in reality shorter pulses are favored because they give rise to a narrower distribution $P(\alpha_0)$ of the rotation angles $\alpha_0 = g\tau_0$. This physics leads to a single optimal pulse of duration slightly shorter than $\tau = \pi/\bar{g}$ and a series of suboptimal pulse durations corresponding to local minima of the average fidelity [see Fig. 8(c)].

Next, we consider the simplest scenario with feedback, which corresponds to $N = 2$ time steps, i.e., feedback on a single measurement. In this case, the control parameters are the first pulse duration τ_0 and the two conditional durations of the second pulse, $\tau_1(m_0 = 1)$ and $\tau_1(m_0 = -1)$. If we use our look-up-table approach to directly optimize these control parameters, the underlying optimization landscape is a 3D function. In Fig. 8(d), we show three 2D cuts of the optimization landscape along with the evolution of the training parameters for 26 feedback-GRAPE training

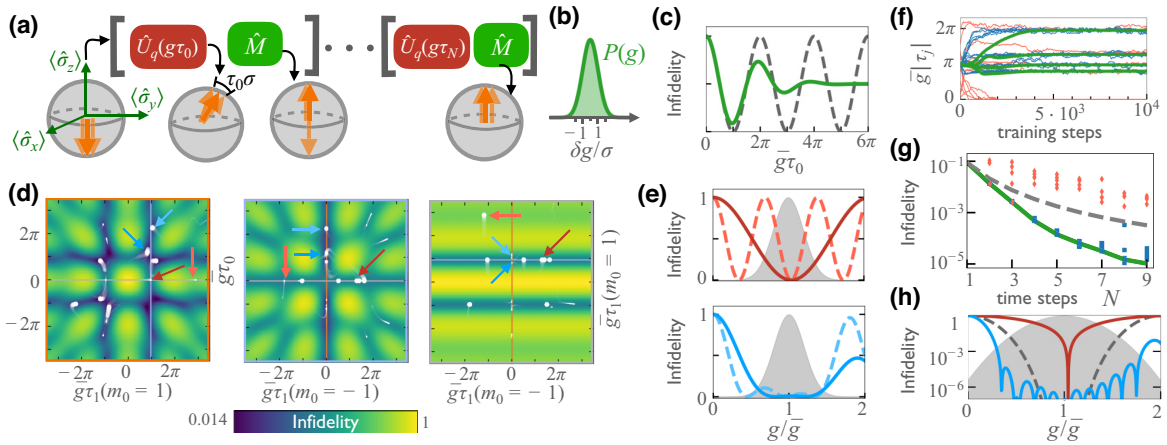

FIG. 8. State preparation with parameter uncertainty. (a) A sketch of the feedback-control sequence consisting of N pulses of duration τ_i interleaved with projective measurements. We also show the corresponding dynamics of an (inhomogeneous) ensemble of qubits represented as spins on the Bloch sphere. The goal of the feedback strategy is to prepare as many spins as possible in the "up" state. (b) The coupling g to the driving field is assumed to be a random Gaussian variable of width σ. (c) The optimization landscape for the case $N = 1$ and $\sigma/\bar{g} = 0.2$ (solid line) and $\sigma = 0$ (dashed line). (d) Three two-dimensional (2D) cuts of the three-dimensional (3D) optimization landscape for the case $N = 2$ and $\sigma/\bar{g} = 0.2$. The one-dimensional (1D) projection of each 2D cut into the other cuts is marked as a line of the same color as the frame. Also shown (on all cuts) are the evolutions of the three learning parameters during training for 26 different training runs. Each run is represented by 200 snapshots, which in turn are represented by dots of varying size and transparency, to represent the distance from the 2D plane and the time at which they were recorded. (e) The infidelity $1 - \langle \mathcal{F}_N(g)\rangle_m$ as a function of the coupling g for four strategies. Also shown is the distribution of the coupling g. The underlying control parameters are marked by the arrows of the same color in (d). (f) The evolution of the control parameters during training. Shown are the durations τ_j of the jth pulse, conditional on the qubit being in the ground state for all previous measurements, $\tau_j \equiv \tau_j(\mathbf{m}_{j-1})$ with $\mathbf{m}_{j-1} = (1, \ldots, 1)$, for $0 \leq j \leq N-1$. Several training runs are depicted, running feedback GRAPE with random initialization (red lines, one run with $N = 7$ and one run with $N = 6$), an initialization close to the intuitive feedback strategy with durations $\bar{g}\tau_j \approx \pi$ and zero otherwise (blue lines; five runs with $N = 4$), or the constrained optimization protocol (green thick lines, one run with $N = 4$). (g) The infidelity $1 - \langle\langle \mathcal{F}_N(g)\rangle_m\rangle_g$ as a function of the number of time steps N for the three different optimization protocols (for the first two protocols, five runs for each N are shown). Also shown is the infidelity for the intuitive feedback strategy $\bar{g}\tau_j \approx \pi$. (h) The same as (e), here for the optimal strategy without feedback (red) and the optimal (blue) and intuitive (gray dashed) feedback strategies with $N = 8$.

runs, starting from random initial conditions. If we view as equivalent strategies that are connected by transformations of the control parameters that leave the fidelity $\langle \mathcal{F}_N(g)\rangle_{\mathbf{m}}$ invariant, we can associate most runs with just four final feedback strategies (for more details, see Appendix K). Their coupling-dependent infidelity $1 - \langle \mathcal{F}_N(g)\rangle_{\mathbf{m}}$ is displayed in Fig. 8(e). A plurality of the training runs (10 out of 26 runs) converges to the optimal strategy, which consists of the combination of a pulse of duration slightly shorter than $\pi/\bar{g}$ followed or preceded by a slightly longer pulse. This leads to a small infidelity over the full width of the coupling distribution $P(g)$ [cf. the blue continuous line in Fig. 8(e)]. However, for a significant number of runs (9 out of 26), the duration τ_0 of the first pulse converges asymptotically to zero during training [cf. the rightmost cut in Fig. 8(d)]. In other words, the first pulse is switched off, effectively reducing the number of time steps to $N = 1$. The resulting infidelity is small on a much narrower band [cf. the top of Fig. 8(e)]. The training can lead to this type of solution because isolated attractors for the $N = 1$ optimization landscape are promoted into 1D manifolds of attractors in the $N = 2$ optimization landscape [cf. the middle and right-hand cuts in Fig. 8(d) with Fig. 8(c)].

The insight into the learning dynamics gained for the case with $N = 2$ time steps can be transferred to the general case of an arbitrary number of time steps N. For a typical run, one or more pulses are switched off, effectively reducing the number of time steps [cf. the red lines in Fig. 8(f)]. This leads to many runs ending up in suboptimal solutions [cf. the red diamonds in Fig. 8(g)]. We note in passing that we observe a similar learning dynamics for the state preparation of complex superposition of Fock states using the Jaynes-Cummings controls. In that case also, a large number of local extrema for an optimization task with N time steps could be constructed, adding idle time steps to the optimal solution for a smaller number of time steps (see Appendix F). We expect the same type of local extrema to appear in many optimal control tasks that take as ansatz a quantum circuit comprising a finite

sequence of parametrized building blocks, irrespective of whether or not measurements are present.

The challenge posed by the local minima can be addressed by taking a physically motivated initialization of the control parameters, e.g., $\tau_j(\mathbf{m}_{j-1}) \approx \pi/\bar{g}$ if $\mathbf{m}_{j-1} = (1,\ldots,1)$ and $\tau_j(\mathbf{m}_{j-1}) = 0$ otherwise. This is close to a π pulse for $g \approx \bar{g}$ if no previous measurement with outcome $m = -1$ has indicated that the spin has flipped. With this approach, we consistently reach the optimal solution for $N \leq 6$ [cf. the blue squares in Fig. 8(g)]. For even larger N, we reach a regime in which for a typical batch of trajectories used to calculate the gradient, all spins have flipped. This leads to a very noisy gradient, making it difficult to distinguish between many available feedback strategies with low infidelity. We eliminate this problem taking the ansatz $\tau_j(\mathbf{m}_{j-1}) = 0$ if $\mathbf{m}_{j-1} \neq (1,\ldots,1)$ for the feedback protocol. In this way, the feedback task is reduced to the optimization of N control parameters $\tau_j \equiv \tau_j(\mathbf{m}_{j-1})$ with $\mathbf{m}_{j-1} = (1,\ldots,1)$. At the same time, the number of possible measurement outcomes is also reduced to the same value. This reduction of the decision tree to only N branches makes it efficient to evaluate the sum in Eq. (2) without resorting to measurement sampling, reducing the gradient fluctuations. In addition, it eliminates the basin of attraction of the suboptimal strategies with a reduced effective number of pulses. Overall, it leads to very robust learning even for large values of N [cf. the green lines in Figs. 8(f) and 8(g)].

Finally, we comment on the robustness to parameter uncertainty for the feedback strategies obtained using feedback GRAPE. The average infidelity $\langle\langle\mathcal{F}_N\rangle_{\mathbf{m}}\rangle_g$ obtained using the feedback strategy with $N = 8$ time steps is of the order of approximately 10^{-5} [cf. Fig. 8(g)]. This means that, in spite of the broad distribution of couplings g, only one qubit out of every 10^5 would remain in the ground state. This compares to approximately 50 with the intuitive feedback strategy and 10^4 using the optimal strategy without feedback. We note that the infidelity $1 - \langle\mathcal{F}_N(g)\rangle_m$ is suppressed over a broad range of couplings g, remaining below the threshold 10^{-3} in a broad band of width approximately $1.5\bar{g}$ [cf. Fig. 8(h)]. From this, we can conclude that the robustness of our strategy to parameter uncertainty extends beyond the particular coupling distribution used during training.

This example has thus convincingly shown the ability of feedback GRAPE to deal with parameter uncertainties, both by finding strategies that properly take into account the size of the fluctuations and, on top of that, by exploiting the extra information obtained via measurements.

IV. SCALABILITY AND OPTIMIZATION LANDSCAPE IN FEEDBACK GRAPE

As we see in the numerous examples presented so far, feedback GRAPE performs very well for quantum feedback tasks in physically relevant scenarios, including the preparation and stabilization of rather complex states. Even though it occasionally gets stuck in local optima, in our examples this can often be remedied very simply by rerunning from different random starting points a number of times. Nevertheless, in this section, we want to address the aspects of the nonconvex optimization landscape and scaling toward larger quantum systems, such as those consisting of many qubits, in a more general fashion. These challenges are, of course, by no means unique to feedback GRAPE and we consequently rely a lot on observations that have been made in the literature, starting from the original GRAPE and going toward recent results on variational quantum circuits.

In the original GRAPE article [3], it has already been recognized that, generally speaking, GRAPE is a nonconvex optimization problem and it has been suggested that adding stochasticity to the gradient update step could help to jump out of local minima. More refined approaches would perform simulated annealing, i.e., slowly reducing the noise strength over time. We note that, in contrast to GRAPE itself, some form of stochasticity is automatically generated in our case by the random outcomes of measurements. The noise strength can effectively be reduced over time using a learning-rate schedule.

Two common approaches to working around local minima are also demonstrated in our own numerical experiments, reported in the present paper. The first approach consists of avoiding the local minima by starting from a smart initialization of the training parameters. This could be physically motivated as in Sec. III F or, as is sometimes done in RL, obtained using some form of pretraining based on supervised learning [68] of an existing approximate strategy. The second approach consists of modifying the ansatz for the feedback protocol, which will also modify the optimization landscape, possibly eliminating or reducing the problematic local minima. A better ansatz could be found using physical insight, as in Sec. III F, or using some form of derivative-free optimization, e.g., model-free RL as in Ref. [69].

Another option for addressing the issue of local optima is to employ the natural gradient. In Ref. [70], it has been shown, for variational quantum circuits in systems of up to 40 qubits, that this technique, though computationally more expensive, avoids local minima most successfully when compared to the more well-known techniques, i.e., direct adaptive gradient descent or quasi-Newton methods such as the Broyden-Fletcher-Goldfarb-Shanno algorithm. Natural gradient can directly be applied to state-preparation problems by computing the Fubini-Study metric of the final state based on its dependence on the control parameters. Thus, it could be employed to help convergence in feedback GRAPE when the technique is applied to larger qubit numbers.

The examples of feedback-controlled quantum dynamics on which we focus in this paper can be viewed as a combination of parametrized quantum circuits with classical measurements and feedback. When dealing with the question of scalability toward larger qubit numbers, the recent literature on variational quantum circuits (without feedback) suggests that another challenge may arise that goes beyond the generic problem of getting stuck in local optima for nonconvex optimization tasks. As has been recognized first in Ref. [71] and subsequently discussed at length in the literature, one may be stuck in parts of the parameter landscape with essentially zero gradients, i.e., gradients that are exponentially small in the number of qubits; this is the infamous problem of "barren plateaus." Fortunately, the importance of this problem for the quantum computing community has led to a succession of possible suggested solutions, all of which could be transferred to an application such as feedback GRAPE in case the issue arises when applying it eventually to systems with larger numbers of qubits. The proposed solutions comprise: (i) smart parameter initialization [72,73], e.g., choosing parameter values that initially lead to unitary blocks equal to the identity; (ii) being smart in the choice of circuit ansatz but avoiding overparametrization (too expressible ansatz structures) [74]; and (iii) constructing a cost function from local observables instead of a global cost function such as the fidelity [75,76]. Last and most relevant to our work, it has recently been shown that incorporating in the quantum circuit the same type of stochastic local measurements that are also used in feedback GRAPE could by itself induce a phase transition to a regime without barren plateaus [77]. While in their work the measurements only introduce decoherence, it would be worthwhile to explore whether they also help to avoid such plateaus in true feedback scenarios.

Beyond these aspects of the optimization landscape, the overall performance of feedback GRAPE, and hence its scalability, is also governed by the computational cost associated with single trajectories. Just like any other model-based method, our method can deal only with systems the time evolution of which can be efficiently simulated on a classical computer. The computational cost of calculating the gradient through the system dynamics grows linearly with the number of time steps, just like the cost of the direct time evolution itself. Furthermore, when addressing the scalability for multiqubit systems, it is true that the cost of a simulation will increase with the size of the Hilbert space and thus rise exponentially with the number of qubits. However, this scaling is no worse than in the original GRAPE or for any other model-based RL method, including those methods that are based on simulations interacting with a model-free approach. From experience with numerical simulations of multiqubit systems, we deem feedback GRAPE to still be feasible up to about ten qubits when simulating master equations and maybe 20 qubits when resorting to quantum jump approaches, evolving pure states. This already covers a lot of unexplored territory for quantum feedback.

Another important aspect is the number of trajectories needed for convergence toward the optimal strategy. This is extremely scenario dependent and therefore hard to predict in general. However, empirically we have seen that, typically, thousands of time-evolution trajectories are to be evaluated to converge to an optimum. It is here that the model-based approach of feedback GRAPE has a big advantage over model-free approaches, since the latter have to employ a lot more trajectories just to implicitly learn the expected behavior of the quantum system. Indeed, among our examples we have briefly discussed the superior performance of GRAPE versus a model-free RL approach in the case of SNAP-gate-based cavity-state preparation.

We mention in passing another advantage of feedback GRAPE: it does not require a real-time Bayesian estimate of the quantum state during deployment in an experiment. In this sense, it is more scalable than other existing model-based quantum feedback approaches based on so-called Bayesian quantum feedback.

Finally, another aspect that may affect scaling is generalizability. For certain feedback tasks, our method allows a generalization of the feedback strategy. This is exemplified by our stabilization of a kitten state for a long sequence of 200 measurements, 20 times longer than the sequences seen during training. This generalization power substantially decreases the computational cost expended during training.

V. EXTENSIONS

Before concluding our discussion, we outline possible extensions of the general feedback-GRAPE technique introduced above.

A. Reducing sampling noise by using a value function

The average of the return over different measurement outcome sequences is obtained by sampling, which introduces noise into the estimate of the gradients. We can help to suppress the noise by adopting value-function approaches that are known as a general technique in reinforcement learning [30].

To start, we need to discuss the structure of the rewards more carefully. Above, we introduce the overall return (cumulative reward) as the quantity to be optimized. We can also assign the rewards more specifically to individual time steps. For example, during state stabilization, we can evaluate the fidelity at each time step and sum it over time to obtain the return. Likewise, it is customary in some optimal control settings to punish large control amplitudes at any given time step. In all these cases, the return is a sum $R = \sum_{j=1}^{N} r_j$ of individual rewards.

More precisely, in the original approach, we simply set $R = r_1(m_1|\theta) + r_2(m_2, m_1|\theta) + \ldots$. Here, $r_j(m_j, m_{j-1}, \ldots|\theta)$ is the instantaneous reward obtained after time step j (which consists of some control, some measurement yielding m_j, and possibly a further control step before assigning the reward). For any time step j, this then yields two contributions to the overall gradient-ascent update. For example, at $j = 2$ in a given trajectory with randomly sampled $m_1, m_2, \ldots$, we obtain the following contributions:

$$\partial_\theta r_2(m_2, m_1|\theta) + \partial_\theta \ln P(m_2|m_1, \theta) \times \{r_1(m_1|\theta) + r_2(m_2, m_1|\theta) + r_3(m_3, m_2, m_1|\theta) + \ldots\}. \tag{5}$$

Adding up these contributions for all j and averaging over trajectories, we recover Eq. (3).

This is a Monte Carlo sampling approach. One concern in any such approach is the sampling noise, i.e., in our case the fluctuations of the quantity shown above between different trajectories. We can now take inspiration from the domain of model-free reinforcement learning and the general theory of reinforcement learning [30], where approaches have been invented to reduce the variance in estimations of the gradient update. Recall that in our case, the variance stems from the stochasticity of measurements, whereas in model-free RL it stems from the stochasticity of policy action choices that is encountered in policy-gradient and actor-critic approaches, plus any stochasticity of the environment dynamics. Even though the following steps follow the corresponding tricks known in the model-free RL community very closely, we display them explicitly here, for our modified scenario. This should help avoid any confusion and make this presentation self-contained.

First, when evaluating the gradient above, we need only include the sum of future rewards, since only those can be influenced by the present measurement result. In the example of Eq. (5), this means that the term $r_1(m_1|\theta)$ on the second line may be dropped, as it is independent of m_2, i.e., the new measurement result. Mathematically, this follows because when we eventually perform the average over trajectories, we have to multiply Eq. (5) by $P(m_2, m_1|\theta) = P(m_2|m_1, \theta)P(m_1|\theta)$. Collecting terms, the m_2 dependency for the r_1 contribution ends up in a sum $\sum_{m_2} \partial_\theta P(m_2|m_1, \theta)$. This sum turns out to be zero due to the normalization of the conditional probability for any value of θ. This insight holds for any j, where it is used to drop all r_k $(k < j)$ when they multiply $\partial_\theta \ln P(m_j|m_{j-1}, \ldots, \theta)$.

Second, to further suppress stochastic fluctuations, one can learn a value function V, which is a function of the current state and represents the expected future cumulative reward, averaged over all possible future measurement outcomes. Thus $V(m_j, m_{j-1}, \ldots|\theta)$ is defined to be

$$E(r_{j+1} + r_{j+2} + \ldots|m_j, m_{j-1}, \ldots, \theta),$$

where the label E denotes the expectation value over future rewards, conditioned on the preceding measurement results.

Typically, V would be expressed as an NN, though a look-up table can also be used in the case of a modest number of discrete measurements. The input to the value network would be some representation of the current "state" s. This state could be identified directly with the sequence of previous measurement results, as indicated in our notation above, $s_j = m_j, m_{j-1}, \ldots$ (which uniquely determines the current state). Alternatively, this state could also be represented by some version of the current quantum state (e.g., the density matrix), if that proves easier to handle for the network. The value network would be trained to output the expected (averaged) future cumulative reward, counted from this state onward. The value training would proceed in the fashion known from general reinforcement learning, i.e., using the Bellman update equation [30] $V^{\text{new}}(s_j) = V(s_j) + \alpha(r_j + \gamma V(s_{j+1}) - V(s_j))$, where $\alpha < 1$ is some update factor and $\gamma \leq 1$ is some discount factor to reduce the weight of long-term rewards ($\gamma \to 1$ in the ideal case discussed up to now). When using an NN, V^{new} would be the new target value for the value network during a supervised-learning update. Once an approximation to the value function has been learned in this manner, we can proceed as in advantage actor-critic approaches to model-free RL. This means that in the gradient-ascent procedure of the feedback-GRAPE approach, one would replace the (future) return by the *advantage* $A_j = r_j + \gamma V(s_{j+1}) - V(s_j)$, which expresses the improvement over the currently expected future return. In effect, this reduces the variance of the gradient estimates by subtracting a convenient baseline, without changing the average gradient update.

Concretely, Eq. (5), the gradient contribution from time step $j = 2$, would be replaced by the following:

$$\partial_\theta r_2(m_2, m_1|\theta) + \partial_\theta \ln P(m_2|m_1, \theta) \times \{r_2(m_2, m_1|\theta) + \gamma V(m_2, m_1|\theta) - V(m_1|\theta)\}. \tag{6}$$

The first line is unchanged but in the second line r_1 is dropped, as explained before. Moreover, the sum of $r_3 + r_4 + \ldots$ is replaced by $\gamma V(m_2, m_1|\theta)$, which is the expectation of the future return (such that averaging over $m_3, m_4, \ldots$ has already been carried out, reducing sampling noise). Finally, $V(m_1|\theta)$ is subtracted, to further reduce the variance by canceling the expected value, given m_1. This is possible for the same reason that we could drop $r_1(m_1)$, as explained above. The extension to arbitrary $j \neq 2$ is obvious.

In summary, such an enhanced feedback-GRAPE method would run trajectories with deterministic continuous controls and stochastic discrete quantum measurements just as before. However, it would learn a value function to represent expected future returns and it would use

that value function to modify the gradient-ascent procedure and reduce fluctuations.

B. Multitarget quantum feedback control

Whenever we are employing NNs to represent the feedback-based controls, a straightforward but powerful extension of feedback GRAPE suggests itself. We may feed a representation of a variable target state Ψ (or, in general, the target task, however it is defined) into the network: $F_j(\theta_j, m_j, \ldots; \Psi)$. The whole feedback-control strategy is then trained on many different randomly chosen tasks (e.g., many possible target states).

Such approaches have been successful recently for other control challenges, e.g., they are being investigated in robotic navigation and the general field of multitarget reinforcement learning [78,79]. Multitarget schemes have also been recently suggested to improve variational quantum circuits [80]. The benefit is data efficiency: the network learns to generalize from the training tasks to other similar tasks, which requires less overall effort than to retrain a freshly initialized network for each task.

VI. CONCLUSIONS AND OUTLOOK

In this work, we present a general scheme for the direct gradient-based discovery of quantum feedback strategies. This scheme, which we term "feedback GRAPE," works for arbitrarily strong (discrete or continuous) nonlinear stochastic measurements, which so far has been possible only using the less-data-efficient approaches of model-free reinforcement learning.

We observe very good performance, significantly beyond the state of the art, when testing the method on a challenging set of feedback tasks in an important practically relevant quantum optical scenario. Overall, our method opens up a new route toward solving challenging feedback-based control tasks, including tasks in quantum communication and quantum error correction on multiqubit or qubit-cavity systems. Besides presenting and analyzing the basic approach, we also discuss extensions such as advantage functions (for reducing sampling noise) and training on multiple targets (to increase data efficiency and exploit transfer learning).

ACKNOWLEDGMENTS

The research is part of the Munich Quantum Valley, which is supported by the Bavarian state government with funds from the Hightech Agenda Bayern Plus.

APPENDIX A: BRIEF RECAP OF REINFORCEMENT LEARNING

For the reader with a physics or optimal control background, we very briefly recall some key concepts in reinforcement learning (RL). However, these remarks serve only to provide additional context and are not necessary to understand the feedback-GRAPE technique introduced in the main text.

The term RL covers a set of techniques for discovering optimal control strategies, typically involving feedback [30]. The setting can always be phrased as an agent, i.e., a controller, interacting with an environment, where the latter may represent, e.g., a device to be controlled. The goal is always to discover a good strategy for the agent, by optimizing some reward—e.g., a fidelity, in the quantum setting. A first distinction is between model-based approaches, which require and exploit a simulation of the environment, and model-free approaches, where the environment is treated as a black box and the agent only learns implicitly about the behavior of the environment through repeated training runs. Feedback GRAPE would be classified under the domain of model-based approaches, while most general RL algorithms used in machine-learning applications nowadays are model free. A subcategory, sometimes leading to confusion, consists of those cases where model-free algorithms are used to train an agent *in silico*, i.e., on simulated environments.

While we explain feedback GRAPE in depth in the main text, here we briefly comment on one of the two main classes of model-free algorithms, namely policy-gradient approaches, since we briefly compare and contrast some aspects of those against feedback GRAPE in the main text. In such approaches, one represents the policy as a conditional probability to choose an action a given an observed state s of the environment: $\pi_\theta(a|s)$ in standard notation of the field. Here, θ is a set of parameters that will be updated during training. Training proceeds by performing gradient ascent on the cumulative reward R, in the form

$$\delta\theta = \eta \frac{\partial E[R]}{\partial \theta} = \eta \sum_t E\left[R \frac{\partial \ln \pi_\theta(a_t|s_t)}{\partial \theta}\right]. \tag{A1}$$

Here, t is the time step, s_t and a_t are the sequence of states and actions in a particular trajectory, R is the cumulative reward for that trajectory, and E denotes the expectation value over many trajectories. We note in the main text that the logarithmic derivative appearing here relates to the probability of the actions of the agent, whereas a superficially similar logarithmic derivative appearing in feedback GRAPE relates to quantum measurement probabilities, i.e., a property of the environment and not the agent.

APPENDIX B: ALGORITHMIC FLOW-CHART REPRESENTATION OF FEEDBACK GRAPE

In Fig. 9, we represent the working flow of feedback GRAPE for the special case of discrete measurement outcomes as an algorithmic flow chart. This representation provides more detail than the conceptual representations in the main text.

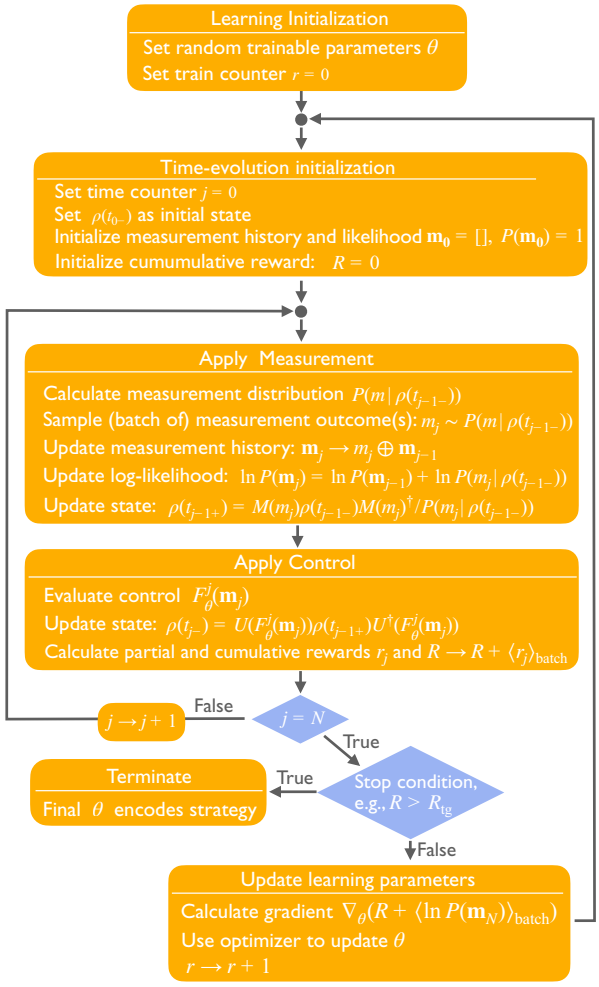

FIG. 9. An algorithmic flow-chart representation of Feedback GRAPE. The measurement outcome m_j at time step j and all other quantities that depend on it have an additional batch dimension that is not explicitly indicated. The symbol $\oplus$ denotes concatenation. Examples of stop conditions include the reward R having to be larger than a target value R_{tg}, $R > R_{\text{tg}}$, or the number of training iterations r having reached a maximum value r_{Max}, $r = r_{\text{Max}}$. The controls $F_\theta(\mathbf{m}_j)$ can be evaluated using an RNN or directly read out from a look-up table.

APPENDIX C: EVALUATION OF THE PARAMETER GRADIENTS OF THE TIME-EVOLVING QUANTUM STATE

In the numerical results in the main text, we employ automatic differentiation to evaluate gradients with respect to the trainable parameter vector θ. This approach is very convenient using modern machine-learning tools. However, alternatively, it is also possible to directly work out analytical formulas to evaluate such gradients, based on our knowledge of the evolution equations. In a particular scenario, where the entries of the vector of trainable parameters θ directly correspond to the controls at different time points, this then produces a suitable extension of the approach advocated in the original GRAPE article [3].

In the following formulas, we assume for simplicity unitary evolution outside the measurements but the extension to (Markovian) dissipative dynamics is comparatively straightforward (using a Liouvillian superoperator instead of the Hamiltonian).

We first describe a general approach that works for any arbitrary choice of the parametrization θ. Further below, we then specialize to a scenario where the original GRAPE idea for efficient gradient evaluation can be applied.

The general task is to obtain the gradient of the quantum state with respect to the trainable parameters θ that enter the controls (and, likewise, the gradient of the final probability $P(\mathbf{m})$ of a measurement sequence $\mathbf{m}$).

In modern machine-learning language, tracking the evolution of parameter gradients in the manner described in the following is connected to the recent development of neural ordinary differential equations [53], where efficiency is obtained by not using automatic differentiation as a black box but, rather, by evaluating analytically the form of the equations of motion for the gradients (and then solving those equations numerically with any efficient solver that is available). We can obtain the parameter gradient of the quantum state by solving the following evolution equation during measurement-free time intervals:

$$i\partial_t \partial_\theta \hat{\rho} = [\partial_\theta \hat{H}, \hat{\rho}] + [\hat{H}, \partial_\theta \hat{\rho}], \tag{C1}$$

where $\hat{\rho}$ is the solution to the original equation of motion, $i\partial_t \hat{\rho} = [\hat{H}, \hat{\rho}]$, and the initial condition at time 0 would be $\partial_\theta \hat{\rho} = 0$ (we set $\hbar \equiv 1$ for brevity). The interesting step now happens at a measurement, where $\hat{\rho}(t^+) = \hat{M}(m)\hat{\rho}(t^-)\hat{M}^\dagger(m)/P_m$, with the probability for the measurement outcome, $P_m = \text{tr}[\hat{M}(m)\hat{\rho}\hat{M}^\dagger(m)]$. For brevity, we suppress the index j (used in the main text) that would indicate the number of the measurement in the sequence. It now follows that we have

$$\begin{aligned}\partial_\theta \hat{\rho}(t^+) = {} & \hat{M}(m)\partial_\theta \hat{\rho}(t^-)\hat{M}^\dagger(m)/P_m \\ & - \hat{\rho}(t^+)\text{tr}[\hat{M}(m)\partial_\theta \hat{\rho}(t^-)\hat{M}^\dagger(m)]/P_m.\end{aligned} \tag{C2}$$

Here, the required $\partial_\theta \hat{\rho}(t^-)$ is the outcome of solving the previous continuous evolution equation up until time t. After this update, the continuous evolution of $\partial_\theta \hat{\rho}(t)$ will proceed. We note, however, that the controls (embedded inside $\hat{H}$ in the present setup) will now depend on the measurement outcome m that was selected. Likewise, for later time intervals, they will depend on the whole previous sequence, as described in the main text.

At the end, we also need the gradient of the extra term, the log-likelihood of the whole measurement sequence,

$\ln P(m_1, m_2, \ldots)$. One way to obtain this is to evolve an unnormalized version of the quantum state, $\tilde{\hat{\rho}}$, the trace of which will give P, which follows the same evolution as the quantum state itself but without the normalization factors that are the probabilities for the individual measurement outcomes. The θ gradient of this unnormalized state again follows an evolution equation of the form shown in Eq. (C1) but with $\tilde{\hat{\rho}}$ substituted for $\hat{\rho}$, during the unitary evolution intervals. However, at a measurement-induced update, we obtain the simpler rule $\tilde{\hat{\rho}}(t^+) = \hat{M}(m)\tilde{\hat{\rho}}(t^-)\hat{M}^\dagger(m)$ and consequently $\partial_\theta \tilde{\hat{\rho}}(t^+) = \hat{M}(m)\partial_\theta \tilde{\hat{\rho}}(t^-)\hat{M}^\dagger(m)$.

What we have described here so far uses fewer assumptions than GRAPE, because the vector of trainable parameters θ can enter the controls in an arbitrary manner. In GRAPE [3], an additional assumption is used to simplify the gradients further and gain efficiency: the components of the vector of trainable parameters θ are supposed to directly correspond to the control values applied at different time steps. That is, schematically speaking, we would have $\theta_1, \theta_2, \ldots$ associated with the controls at time steps $j = 1, 2, \ldots$. This then leads to a further simplification in the evaluation of the gradients. Importantly, if the number of parameters scales with the number of time steps N, then this approach has a run time that grows only linearly in N, while the general approach outlined above would need N^2 operations.

Let us briefly recall the GRAPE approach to gradient evaluation [3], before extending it. In the simplest possible version, with unitary evolution, let us consider the fidelity $\mathrm{tr}(\hat{\sigma}(T)\hat{U}(T,0)\hat{\rho}(0)\hat{U}(0,T))$. The derivative with respect to parameter θ entering the Hamiltonian will produce a contribution for each time $t \in (0, T)$ in the evolution. Specifically, the contribution from time t will be an expression of the type $\mathrm{tr}(\hat{\sigma}\hat{U}(T,t)[-i\frac{\partial \hat{H}}{\partial \theta}, \hat{\rho}(t)]\hat{U}(t,T))$. Using the cyclic property of the trace, this can be reordered to obtain $\mathrm{tr}(\hat{U}(t,T)\hat{\sigma}(T)\hat{U}(T,t)[-i\frac{\partial \hat{H}}{\partial \theta}, \hat{\rho}(t)])$. This can now be reinterpreted, namely, as the overlap between a backward-evolved target state $\hat{\sigma}(t) = \hat{U}(t,T)\hat{\sigma}(T)\hat{U}(T,t)$ and the perturbation of the forward-evolved state at time t: $\mathrm{tr}(\hat{\sigma}(t)[-i\frac{\partial \hat{H}}{\partial \theta}, \hat{\rho}(t)])$.

In machine-learning language, the GRAPE procedure of obtaining gradients in this way can essentially be viewed as an analytically derived version of back-propagation for this specific case of a quantum physical evolution. It is very efficient, since the effort scales only linearly in the number of time steps, even if there is a different independently optimizable parameter $\theta(t)$ for each time step.

The question is how this procedure needs to be modified in the presence of measurements. Let us imagine that we have a particular trajectory with a given fixed sequence of measurement outcomes. We find that we can perform the temporal back-propagation (starting from the final time T) in the same manner as reviewed above, until a point in time $\tilde{t}$ where a measurement has happened (unless, of course, we talk about a time point t later than the last measurement). At that point $\tilde{t}$, we need to replace $\hat{\sigma}(\tilde{t}) = \hat{U}(\tilde{t}, T)\hat{\sigma}\hat{U}(T, \tilde{t})$ by the following expression:

$$\hat{\sigma}'(\tilde{t}) = \frac{1}{P}\hat{M}^\dagger\hat{\sigma}(\tilde{t})\hat{M} - \frac{1}{P^2}\hat{M}^\dagger\hat{M}\mathrm{tr}(\hat{M}^\dagger\hat{\sigma}(\tilde{t})\hat{M}\hat{\rho}(\tilde{t})). \quad \text{(C3)}$$

Here, we define, for brevity, the measurement operator $\hat{M} \equiv \hat{M}_{\tilde{m}}$ at time point $\tilde{t}$, with measurement outcome $\tilde{m}$, and the associated probability $P \equiv P_{\tilde{m}} = \mathrm{tr}(\hat{M}_{\tilde{m}}\hat{\rho}(\tilde{t})\hat{M}_{\tilde{m}}^\dagger)$, where $\hat{\rho}(\tilde{t})$ is already conditioned on previous measurement outcomes, for times less than $\tilde{t}$ and has been obtained by the forward evolution starting from time 0 (with measurements and renormalization of the state after each measurement).

After this procedure has been implemented for the measurement at $\tilde{t}$, we would proceed with the backward evolution of $\hat{\sigma}$ until point t, where the derivative is to be evaluated. There, we would employ the same formula as in the usual GRAPE approach, i.e., we would evaluate $\mathrm{tr}(\hat{\sigma}(t)^\dagger[-i\frac{\partial \hat{H}}{\partial \theta}, \hat{\rho}(t)])$.

If there are multiple measurements between t and T, the backward evolution will proceed by alternating unitary evolution and applying the formula in Eq. (C3).

If we want to treat the unnormalized quantum state in the same manner, e.g., for obtaining the log-likelihood term, we only need the trace of that unnormalized state $\tilde{\hat{\rho}}$ at the end of the time evolution (see our discussion above). Formally, this is as if we were to calculate the fidelity against a state $\hat{\sigma}(T) = 1$, which is given by the identity matrix. We can now evolve this state backward in the manner discussed above but in addition, Eq. (C3) can be simplified: one needs to drop the second term and also formally set $P = 1$ in the first term.

Finally, we briefly remark on how the procedure will change if we are dealing with continuous measurement outcomes (strong continuous measurements, as briefly discussed in the main text, using the "reparametrization trick"). In that case, we do not need the log-likelihood term. However, we now do need to differentiate the measurement outcome $m = f_{\hat{\rho}}^{-1}(z)$, which depends on some random variable z (of a fixed distribution, not dependent on θ) and the quantum state $\hat{\rho}$ (which does depend on θ). As a consequence, Eq. (C3) needs to be modified. We have to add the following terms to the right-hand side:

$$\frac{1}{P}\partial_\theta(\hat{M}^\dagger\hat{\sigma}(\tilde{t})\hat{M}) - \frac{1}{P^2}\mathrm{tr}(\hat{M}^\dagger\hat{\sigma}(\tilde{t})\hat{M}\hat{\rho}(\tilde{t}))\partial_\theta(\hat{M}^\dagger\hat{M}). \quad \text{(C4)}$$

Here, ∂_θ in both parts of this expression is supposed to act only on the $\hat{M}^\dagger$ and $\hat{M}$ terms. This derivative is to be applied in the way $\partial_\theta \hat{M}(m) = (\partial_m \hat{M}(m))(\partial_\theta m)$, where

the derivative of m with respect to θ must be evaluated using the dependence of the inverse cumulative distribution function on the θ-dependent quantum state at that time point.

APPENDIX D: LAW-EBERLY ALGORITHM

As a benchmark with an analytical solution (but still without feedback), we consider the task of preparing an arbitrary pure-cavity state in a cavity-qubit system. This can be achieved by exploiting the well-known Law-Eberly protocol [58]. This algorithm relies on the essential assumption that we start from the ground state. We briefly review it below.

The Hamiltonian that describes the system is a Jaynes-Cummings model with controllable couplings:

$$\hat{H}(t) = (A(t)\hat{\sigma}_+ + A^*(t)\hat{\sigma}_-) + (B(t)\hat{a}\hat{\sigma}_+ + B^*(t)\hat{a}^\dagger\hat{\sigma}_-), \tag{D1}$$

where the first term corresponds to the qubit drive and the second to the cavity-qubit interaction. The two complex controls $A(t)$ and $B(t)$ can assume continuous values.

Law and Eberly uses the particular ansatz $A(t) = 0$ if $B(t) \neq 0$ and vice versa. In this scenario, the dynamics can be viewed as being subdivided into a discrete number of steps N, with each step consisting of one qubit excitation gate, $\hat{U}_q(\alpha_j) = \exp[-i(\alpha_j\hat{\sigma}_+ + \alpha_j^*\hat{\sigma}_-)]$, followed by one cavity-qubit interaction gate, $\hat{U}_{qc}(\beta_j) = \exp[-i(\beta_j\hat{a}\hat{\sigma}_+ + \beta_j^*\hat{a}^\dagger\hat{\sigma}_-)]$. [For given $A(t)$ and $B(t)$, α_j and β_j can be easily obtained by integrating over the relevant time interval.] Since the excitations can be added only one by one via the qubit drive, one can further refine the ansatz assuming that the number of steps N is equal to the maximum number of excitations in the target state, $|\psi\rangle_{\text{target}} = \sum_{n=0}^{N} c_n |n, g\rangle$. To summarize, the goal is to find the parameters $\{\alpha_j\}$ and $\{\beta_j\}$ that solve

$$|\psi\rangle_{\text{target}} = \hat{U}|0, g\rangle, \tag{D2}$$

with

$$\hat{U} = \hat{U}_{qc}(\beta_N)\hat{U}_q(\alpha_N)\hat{U}_{qc}(\beta_{N-1})\hat{U}_q(\alpha_{N-1}) \times \ldots \hat{U}_{qc}(\beta_1)\hat{U}_q(\alpha_1). \tag{D3}$$

The Law-Eberly idea is to start from the target state and progressively remove excitations from the cavity until it becomes empty. In other words, one focuses on the time-reversed time evolution

$$|0, g\rangle = \hat{U}^\dagger |\psi\rangle_{\text{target}}, \tag{D4}$$

with

$$\hat{U}^\dagger = \hat{U}_q^\dagger(\alpha_1)\hat{U}_{qc}^\dagger(\beta_1)\ldots\hat{U}_q^\dagger(\alpha_N)\hat{U}_{qc}^\dagger(\beta_N), \tag{D5}$$

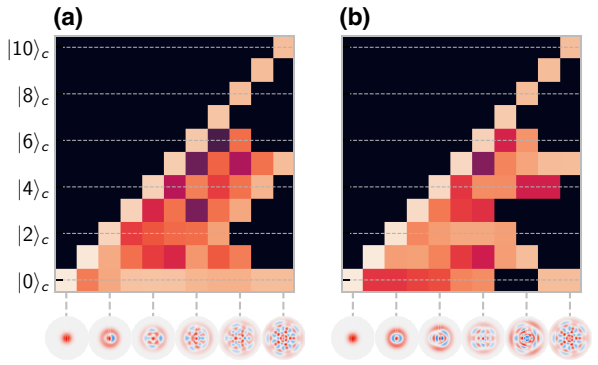

FIG. 10. A comparison between one solution obtained analytically from the Law-Eberly protocol (a) and a strategy found by using gradient ascent (b). The target state is $|\psi\rangle = (|0\rangle + |5\rangle + |10\rangle)/\sqrt{3}$. Even though some details look different, we verify that the gradient-ascent strategy is a valid alternative solution for the Law-Eberly equations (which do not determine the controls uniquely).

and recursively (for decreasing j starting from $j = N$) find the β_j and α_j imposing the conditions $\langle j, g|\psi_j\rangle = 0$, and $\langle j-1, e|\psi_j\rangle = 0$, with

$$|\psi_j\rangle = \hat{U}_q^\dagger(\alpha_j)\hat{U}_{qc}^\dagger(\beta_j)\ldots\hat{U}_q^\dagger(\alpha_N)\hat{U}_{qc}^\dagger(\beta_N)|\psi_{\text{target}}\rangle$$

being the state after $N + 1 - j$ time steps of the time-reversed evolution. These conditions are enforced by the complex nonlinear equations

$$\begin{aligned}
&\langle j, g|\psi_{j+1}\rangle \cos(|\beta_j|\sqrt{j})(\beta/|\beta|) \\
&\quad + i\langle j-1, e|\psi_{j+1}\rangle \sin(|\beta_j|\sqrt{j}) = 0, \\
&\langle j-1, e|\hat{U}_{qc}^\dagger(\beta_j)|\psi_{j+1}\rangle \cos(|\alpha_j|)(\alpha^*/|\alpha|) \\
&\quad + i\langle j-1, g|\hat{U}_{qc}^\dagger(\beta_j)|\psi_{j+1}\rangle \sin(|\alpha_j|) = 0,
\end{aligned} \tag{D6}$$

with $|\psi_{N+1}\rangle \equiv |\psi_{\text{target}}\rangle$ for the first iterative step (corresponding to $j = N$).

It should be noted that the solution of these equations is not unique. This is why Fig. 10 shows two different strategies for the same task, although both of them fulfill the Law-Eberly ansatz.

APPENDIX E: MODEL-FREE REINFORCEMENT LEARNING FOR THE JAYNES-CUMMINGS SCENARIO

It turns out that state-of-the-art model-free RL has surprising difficulties in addressing a physical scenario as important and conceptually simple as the Jaynes-Cummings model. In this appendix, we provide some more details.

We only consider the (simpler) no-feedback case, meaning that only the controls α_j and β_j (see the main text) are available. Since model-free RL already has severe problems in this case, we do not explore the more challenging cases further.

In our numerical experiments, we rely on the RL library Stable Baselines [81], which implements many of the most well-known optimized state-of-the-art RL algorithms. The RL environment (not to be confused with a "physical" environment) has been implemented in the following way:

(a) Action a_j. The two continuous controls, α_j and β_j.
(b) State s_j (i.e., input to the agent). In principle, the no-feedback task requires no state input. However, we choose to make it easier for the agent by supplying the full current quantum state of the system at time t_j. Since the state is pure and the system is closed, we simplify the observation by only using the state vector $|\psi_j\rangle$ (instead of the density matrix). Since it is complex valued, we split its real and imaginary parts and so we have a vector of length $2N$, where N is the size of the Hilbert space.
(c) Reward r_j. The fidelity at step t_j (in various versions, see below).

We use a variety of different approaches to solve the task of pure-state preparation. These include: using either a sparse final reward (i.e., $r_j \neq 0$ only if $j = N$) or else a reward based on the fidelity at each time step; either discrete (discretized) actions or continuous actions; and several different optimization algorithms (PPO [60], A2C [82], HER [83], TRPO [84], and/or DDPG [85]). The results shown in Fig. 3(c) are the best results we can manage to produce among all these approaches. They are obtained with PPO, continuous actions and sparse rewards and using the hyperparameters in Table I.

TABLE I. The hyperparameters for the PPO model-free RL optimization algorithm used in Fig. 3(c). See also the Stable Baselines PPO documentation.

Parameter	Value
GAMMA	0.99
N_STEPS	128
ENT_COEF	0.01
LEARNING_RATE	0.00025
VF_COEF	0.5
MAX_GRAD_NORM	0.5
LAM	0.95
NMINIBATCHES	4
NOPTEPOCHS	4
CLIPRANGE	0.2

APPENDIX F: DETAILS ON LEARNING DYNAMICS USING JAYNES-CUMMINGS OPEN-LOOP CONTROLS

In this appendix, we give more details on the learning dynamics of the open-loop control strategy to prepare a four-component kitten state, $|\psi_{\tilde{\alpha}}^{\mathrm{Kit4}}\rangle \propto \sum_{j=0}^{3} |i^j \tilde{\alpha}\rangle$. This analysis gives a useful insight into the optimization landscape for the open-loop control preparation of complex superpositions of Fock states using Jaynes-Cummings controls.

Our numerical results for $\tilde{\alpha} = 3$, corresponding to the average photon number $\bar{n} \approx |\tilde{\alpha}|^2 = 9$ are summarized in Fig. 11. We preliminarily note that the excitation number of the target state is not bounded in this case. On the other hand, the Law and Eberly protocol allows us to reach only the first N Fock states in N preparation steps. Thus, the optimal strategy will project the target state into the Hilbert space spanned by the first N Fock states and can be obtained using the Law and Eberly algorithm. This procedure also allows us to find the smallest possible infidelity. We choose $N = 20$, corresponding to the minimal infidelity $\mathcal{F} \approx 6 \times 10^{-5}$.

The fidelity as a function of the number of training iterations (or, equivalently, trajectories used during training) for ten different training runs is shown in Fig. 11(a). As should be expected, given that the control parameters slide down a rugged learning landscape, the line shape of the fidelity depends strongly on the initialization. Nevertheless, it displays robust features in the form of a series of plateaus the heights of which do not depend on the initialization. It turns out that the states $\hat{\rho}(t_N)$ prepared following strategies obtained in different runs but corresponding to the same fidelity plateau are also approximately equal. In Figs. 11(b) and 11(c), we show the Wigner function and excitation number distribution $\langle n|\hat{\rho}(t_N)|n\rangle$ obtained with a representative strategy for each plateau in Fig. 11(a). We note that the state is approximately the projection of the target state on a Hilbert space containing the first N_{th} Fock states with the threshold excitation numbers $N_{\mathrm{th}} = 8$, 12, and 16 [cf. Fig. 11(c)]. Importantly, each of these states is prepared using a different strategy in the various training runs [cf. Fig. 11(d)]. Indeed, it is easy to construct several different strategies to prepare exactly these states setting $2(N - N_{\mathrm{th}})$ controls to zero and choosing the remaining $2N_{\mathrm{th}}$ controls to solve the Law and Eberly equation [see Eq. (D6)] for N_{th} preparation steps. It can also be shown that such suboptimal strategies correspond to saddle points of the optimization landscape and that the curvature of the optimization landscape in the direction of increasing fidelity is zero, giving rise to a narrow valley. In this way, we can construct many suboptimal strategies for N preparation steps from an optimal strategy with $N_{\mathrm{th}} \leq N$ steps. Each of these suboptimal strategies corresponds to a narrow valley in the optimization landscape. These valleys

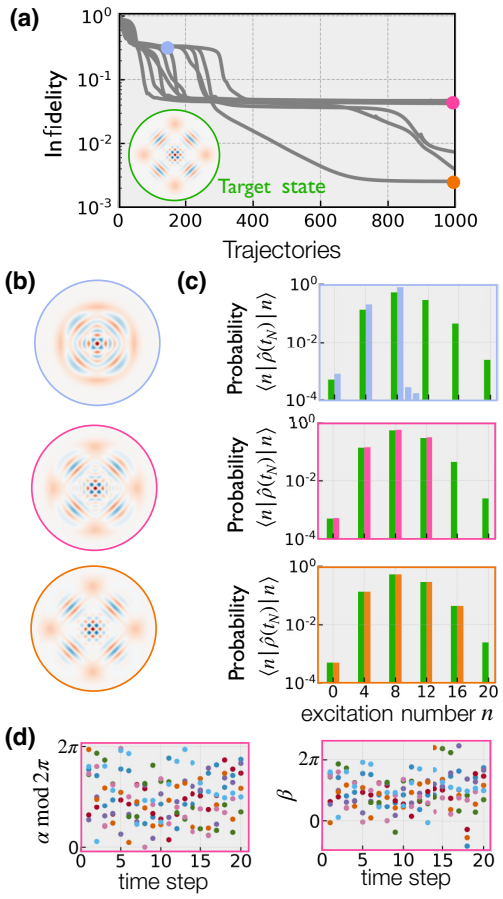

FIG. 11. The preparation of a four-component kitten state with average excitation number $\bar{n} = 9$, as in Fig. 3(d) of the main text. (a) The infidelity as a function of the number of trajectories seen during training for ten training runs with different initial conditions. The Wigner function of the target state is shown as an inset. (b),(c) The Wigner function (b) and the excitation-number distribution (c) after state preparation for three different control strategies discovered during training. The fidelity and number of trajectory used during training are marked in (a) as dots of the same color as the frames in (b) and (c). In (c), the target distribution is also shown in green. The use of seven different training runs leads to approximately the same fidelity after 1000 training iterations [cf. the pink dot in (a)]. (d) The corresponding underlying strategies. For each strategy, the controls as a function of the time step are shown as dots of the same color. Each of these strategies leads to the preparation of approximately the same state, displayed in the second row of (b) and (c).

can cause the training to stall in a suboptimal solution. In fact, we are unable to recover an optimal solution in any of the ten training runs, each comprising 1000 training iterations. Nonetheless, we are able to obtain very good-quality solutions. The best solution we obtain (in one out of ten runs) allows us to prepare the oscillator in the target state projected onto an Hilbert space with cut off $N_{\mathrm{th}} = 16$ [third row in Figs. 11(b) and 11(c)]. This corresponds to the last valley before reaching an optimal solution. We also run a set of ten simulation runs, also comprising the same number of learning iterations but with a larger number of preparation time steps ($N = 28$). In this set of runs (not shown), we reach the fidelity plateau corresponding to $N_{\mathrm{th}} = 16$ more consistently and can even reach the plateau for $N_{\mathrm{th}} = 20$.

In conclusion, our analysis indicates that the optimization landscape for the preparation of complex superpositions of Fock states using open-loop Jaynes-Cummings controls features a very large number of narrow valleys. In this setting, it is crucial to have direct access to the landscape gradient to be able to slowly but steadily slide down the optimization landscape. Very good-quality solutions can be consistently obtained.

APPENDIX G: STATE PREPARATION WITH SNAP AND DISPLACEMENT GATES

In this appendix, we report our results for the open-loop control state preparation of an oscillator state using the universal control scheme based on a set of interleaved SNAP gates $\hat{S}(\{\varphi_n\})) = \sum_n e^{i\varphi_n}|n\rangle\langle n|$ and displacement gates $\hat{D}(\alpha) = \exp[\alpha\hat{a}^\dagger - \alpha^*\hat{a}]$ [66] [cf. Fig. 12(a)]. In this case, the control parameters are the real and imaginary parts of the phase-space displacement α^j together with the phases φ_n^j for the first N_{SNAP} Fock states (the remaining phases are set to zero). This very same optimization task has been already considered by Sivak *et al.* [38] to demonstrate their model-free optimal control approach. This allows an instructive comparison of our model-based results with results obtained using a cutting-edge model-free approach. In order to facilitate the comparison, we use a similar controller as in Ref. [38], i.e., an RNN that takes as input the time step j (cf. Table IV).

The authors of Ref. [38] have found that the main limiting factor in the way of preparing ever more complex quantum states using their model-free approach is a trade-off introduced by the choice of the hyperparameter N_{SNAP}: It is helpful to increase this hyperparameter to improve the expressivity of the sequence of control gates. At the same time, the resulting increased dimensionality of the control-parameter space can make the training unstable. In order to remain in the regime of efficient training, they choose $N_{\mathrm{SNAP}} = 30$ for $N = 9$ preparation steps. Overall, this corresponds to a 288-dimensional control-parameter space. In their numerical experiments, this trade-off became apparent during the preparation of large finite-energy grid states, $|\psi_\Delta^{\mathrm{GKP}}\rangle = \hat{E}_\Delta \sum_{j\in\mathbb{Z}} \hat{D}(j\sqrt{\pi})|0_x\rangle$, where $|0_x\rangle$ is a position eigenstate localized in the origin and $\hat{E}_\Delta = \exp[-\Delta^2\hat{a}^\dagger\hat{a}]$ is the envelope operator. These states have been proposed by Gottesman, Kitaev, and Preskill [61] to encode logical qubit states in an oscillator Hilbert space. They are

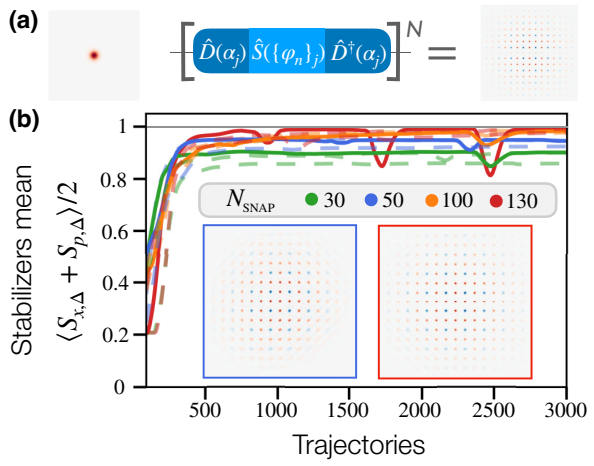

FIG. 12. The state preparation (open-loop control) of a grid state with more powerful controls. (a) The sequence of parametrized controls during a time-evolution trajectory. Each time step consists of two phase-space displacement gates separated by a SNAP gate $\hat{S}(\{\varphi_n\})$. The phases φ_n for $0 \le n < N_{\mathrm{SNAP}}$ are predicted by an RNN that is given the time j as an input (for $n \ge N_{\mathrm{SNAP}}$ $\varphi_n = 0$). Also shown are the Wigner functions of the initial state and of the target grid state. (b) The mean value of the two (finite-energy) stabilizers $\hat{S}_{x,\Delta}$ and $\hat{S}_{p,\Delta}$ as a function of the number of trajectories sampled during training (or, equivalently, the number of training iterations) for four values of N_{SNAP} (three runs each; the best run is displayed as a solid line). The target grid state is in the manifold with $\langle \hat{S}_{x,\Delta} \rangle = \langle \hat{S}_{p,\Delta} \rangle = 1$. The running average over 100 trajectories is plotted. Also shown is the Wigner function after training for $N_{\mathrm{SNAP}} = 50$ and $N_{\mathrm{SNAP}} = 130$. For small N_{SNAP}, the tail of the Wigner function is distorted, while for larger N_{SNAP} it is indistinguishable from the target Wigner function by the naked eye. This indicates that the quality of the strategy is limited by the expressivity of the parametrized control sequence. Compared to model-free RL [38], the model-based approach used here allows us to explore a higher-dimensional parameter manifold (larger values of N_{SNAP}) and, thus, to obtain better quality results for large grid states. The parameters are as follows: $N = 9$ and for the grid state $\Delta = 0.15$ corresponding to $\sqrt{\mathrm{var}(n)} \approx \bar{n} \approx 1/(2\Delta^2) \approx 22$ (the Hilbert space contains 130 Fock states).

eigenstates of the stabilizers $\hat{S}_{x,\Delta} = \hat{E}_\Delta \hat{D}(\sqrt{\pi}) \hat{E}_\Delta^{-1}$ and $\hat{S}_{p,\Delta} = \hat{E}_\Delta \hat{D}(i\sqrt{\pi}) \hat{E}_\Delta^{-1}$ with eigenvalue $+1$. For small Δ, their Wigner function displays a very fine structure in phase space in the form of a large grid of peaks in phase space. The Wigner function of the largest grid state considered in Ref. [38] (corresponding to $\Delta = 0.15$) is shown in Fig. 12(a). In that work, the authors considered as a figure of merit the stabilizer mean value $\langle \hat{S}_{x,\Delta} + \hat{S}_{p,\Delta} \rangle / 2$, which has optimal value $\langle \hat{S}_{x,\Delta} + \hat{S}_{p,\Delta} \rangle / 2 = 1$. They were able to demonstrate a best value of $\langle \hat{S}_{x,\Delta} + \hat{S}_{p,\Delta} \rangle / 2 \approx 0.93$ out of six training runs with $\Delta = 0.15$, which was limited by the expressivity of their control sequence with $N_{\mathrm{SNAP}} = 30$.

In our GRAPE-type approach, due to the direct access to the gradient of the reward, our training remains efficient for much larger values of N_{SNAP} [cf. Fig. 12(b)]. More precisely, we consider N_{SNAP} up to 130 with the same number of preparation time steps ($N = 9$) and the same type of controller (an RNN that takes the time step as input) as in Ref. [38]. This corresponds to a 1188-dimensional control space, much larger than the one that can be handled in the model-free approach. In spite of this larger control-parameter space, training requires only 1% of the trajectories. Most importantly, we obtain a better-quality solution, as reflected by the best stabilizer value of $\langle \hat{S}_{x,\Delta} + \hat{S}_{p,\Delta} \rangle / 2 \approx 0.995$ out of three training runs with $\Delta = 0.15$ and $N_{\mathrm{SNAP}} = 130$.

APPENDIX H: DETAILS ON FEEDBACK-GRAPE ALGORITHM AND ON PHYSICAL SIMULATIONS

As explained in the main text, in the feedback-GRAPE approach presented in this paper, we can produce the control values (conditioned on previous measurement results) either with the help of an NN or with the help of a look-up table (containing trainable control values). In this appendix, we present more details on both of these approaches, as implemented for the specific numerical examples shown in the main text.

In our illustrative physical scenario (the feedback-controlled Jaynes-Cummings model), there are four control parameters: α_j, β_j, γ_j, and δ_j. In the most general case, where arbitrary superpositions should be generated, α_j and β_j need to be complex. In the scenarios the results of which are displayed in the main text, this is not needed due to the nature of the target states. However, we check independently that the whole approach works just as well for complex control parameters.

1. Neural network

We first discuss the case when the controls are computed by means of an NN. This network can receive the measurement results so far, $m_1, m_2, \ldots, m_j$. Alternatively, we can also supply it with the quantum state as input, which has been updated according to the measurement outcomes. Both techniques supply the full information content needed to apply the next control.

For the "state-as-input" approach, we define a fully connected NN that takes the density matrix of the system as input. Since the density matrix is complex valued, we choose to split it into its real and imaginary parts and to stack it in such a way that for a $N_H \times N_H$ density matrix, the input tensor has shape $[N_H \times N_H, 2]$.

The fully connected NN is employed both for the no-feedback case (pure-state preparation), where in principle no such input would be needed (but can still be helpful for convergence) and also for the more interesting feedback cases.

TABLE II. The hyperparameters of the fully connected NN.

Parameter	Value
Neurons	$[[N_H \times N_H, 2]$, Flatten, 30, 30, 2, or 4]
Batch size	1
Activation	ReLU
Initializer	Glorot uniform
Initial-bias last layer	π

If, on the other hand, we want to supply the measurement results directly, then we employ an RNN. For our scenario, its input at each time step is a binary measurement outcome $m_j \in \{-1, +1\}$. When an RNN is used, due to the probabilistic outcome of the trajectories during a simulation, it is useful to feed batches of multiple randomly sampled trajectories as input to the network.

As already mentioned, both types of NN output real-valued controls α_j, β_j, γ_j, and δ_j to be applied in the next time step. When complex-valued controls are required, two additional neurons can be added to the output of the NNs and they correspond to the imaginary parts of α_j and β_j. In the main text, we do not use complex controls, because these are not needed for the tasks considered there.

Our NNs are implemented using KERAS and their hyperparameters are shown for completeness in Tables II–IV.

2. Look-up table

Another way to represent the entire feedback-based control strategy is to use a look-up table, which essentially is just a list of optimizable parameters. In the case of feedback, we have to build a look-up table that encodes the structure of a decision tree. For binary measurement outcomes (as used here), this has $\sum_{n=0}^{N} 2^n$ entries, each of which is the vector of all control parameters, i.e., in our scenario $(\alpha_j, \beta_j, \gamma_j, \delta_j)$. Each column of this table represents the 2^j possible control-parameter vectors at time step $j \in \{0, \ldots N\}$. At $j = 0$, we have only one set of numbers, which stand for the (only) possible control vector to apply (not dependent on any previous measurement; in our case, reduced to only the entries controlling the first measurement). At step $j = 1$, we have two sets of numbers and we apply the set of controls corresponding to the observed measurement and so on and so forth. By doing so, we can apply controls conditioned on the "memory" of all previous measurements, at the cost of keeping an exponentially growing number of entries in the memory of the computer. Many of those will likely not be explored at all, if their probabilities are too small.

TABLE III. The hyperparameters of the RNN for Figs. 5, 15 and 16.

Parameter	Value
Type RNN-cells	GRU
Neurons	[30, 2, or 4]
Batch size	10
Dropout	0.2
Input shape	[BATCH_SIZE, 1, 1]
Activation	tanh
Recurrent activation	Sigmoid
Initializer	Glorot uniform
Initial-bias last layer	π

TABLE IV. The hyperparameters of the RNN used for Figs. 7 and 12. The batch size is 16 and one for Figs. 7 and 12, respectively. The hyperparameter N_{SNAP} is 15 for Fig. 7 and varies from 30 to 130 in Fig. 12 (cf. the inset). For Fig. 12 the input is the time step j, $0 \leq j < N = 9$, while for Fig. 7 it is the measurement outcome m_j.

Parameter	Value
Type RNN-cells	GRU
Neurons	[30, 30, 30, 2+N_{SNAP}]
Input shape	[BATCH_SIZE, 1, 1]
Activation dense layers	ReLU
Recurrent activation	Sigmoid
Initializer	Glorot uniform
Initial-bias last layer	0.1

In our numerical experiments, we go as far as look-up tables containing about $2^{21} \sim 2 \times 10^6$ entries, which is still easily handled. The initial condition for the whole table is to set each parameter value to a random number uniformly distributed within $(0, \pi)$.

For Fig. 8, we use a large batch of size 10 000 (1000) in the approaches with (without) Monte Carlo sampling. For the approach with random initialization, the look-up-table entries $\bar{g}\tau_i(\mathbf{m}_{j-1})$ are uniformly distributed within $(-2\pi, 2\pi)$. In the approach with smart initialization, the initial entries are weakly randomized: for $\mathbf{m}_{j-1} = (1, \ldots, 1)$, we choose $\bar{g}\tau_j(\mathbf{m}_{j-1}) = \pi + z(\mathbf{m}_{j-1})$ with $z(\mathbf{m}_{j-1})$ uniformly distributed in the interval $(0, 1)$; for the remaining measurement outcomes, we set $\tau_j(\mathbf{m}_{j-1}) = 0$.

In several results mentioned in the main text, we use a look-up table "without memory." This means that there is just one control-parameter vector for each step j, instead of a tree-type structure with an exponentially growing number of parameters. Thus, we still optimize the controls but ignore the result of previous measurements. This is used both for the "nonadaptive" scheme for the purification task in Fig. 4(c) and in Fig. 5(g).

A sketch of all of the three feedback-based strategies discussed here and in the main text (NN with state as input, RNN with measurement sequence as input, and a tree-type look-up table) is shown in Fig. 13.

In any case, in whatever ways we choose to parametrize our controls, we have a finite number of parameters that need to be learned. In order to do so, the optimizer

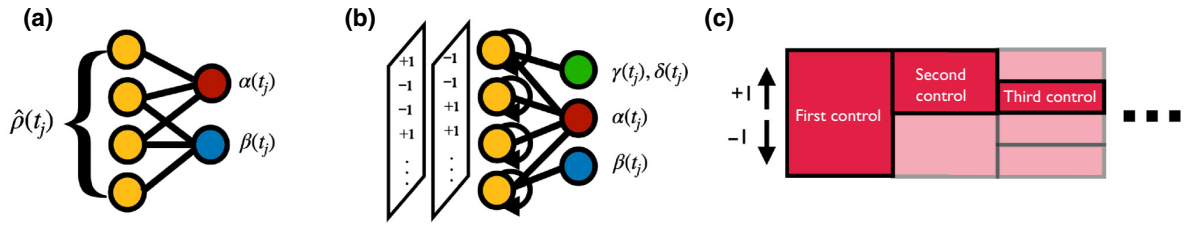

FIG. 13. A sketch of the three alternative types of trainable controls that can be employed in feedback GRAPE. (a) A fully connected NN that receives the density matrix (quantum state) of the system as input and outputs the controls. (b) An RNN with gated-recurrent-unit (GRU) cells as recurrent neurons (input = measurements). (c) A look-up table (input = measurements) with $\sum_{n=0}^{N} 2^n$ entries (when feedback is required and when the measurement outcomes are binary, as shown here), and each entry contains the controls that need to be applied after observing a particular measurement sequence.

employed for every example is ADAM [86] and its hyperparameters are shown in Table V.

3. Physical simulations

In the unitary case, we simply apply the sequence of parametrized unitaries, as explained in the main text. In the case of decay (in the state-stabilization scenario), we solve the master equation for the density matrix during the respective time intervals (where decay is present). Specifically, we simulate the weak Markovian coupling of the oscillator to a zero-temperature bath via the Lindblad master equation,

$$\dot{\hat{\rho}} = \kappa \left(\hat{a}\hat{\rho}\hat{a}^\dagger - \frac{1}{2}\{\hat{a}^\dagger\hat{a}, \hat{\rho}\} \right). \tag{H1}$$

We discretize this continuous time evolution applying the fourth-order Runge-Kutta method.

We choose the Hilbert space to have a finite dimension N_H with a cutoff in the Fock-state excitation number. An appropriate choice of the cutoff depends on both the initial and the target state and ranges from ten to 130 in our simulations.

APPENDIX I: FURTHER NUMERICAL RESULTS

In this appendix, we present a few more numerical results to illustrate various options or aspects of the technique.

TABLE V. The ADAM hyperparameters.

Parameter	Value
LEARNING_RATE	0.01[a]
BETA_1	0.9
BETA_2	0.999
EPSILON	1E-7
CLIPNORM	1
CLIPVALUE	0.5

[a]Unless otherwise specified.

1. Effect of different initial conditions on the training

In order to assess the variability during the training, we show in this subsection how the results of Figs. 4 and 5 can change, depending on the choice of different random initial conditions of the algorithm. As a first example, we show in Fig. 14 a plot equivalent to Fig. 4(c) but in this case we want to differentiate the distinct strategies found by feedback GRAPE. The majority of the adaptive runs can systematically reach higher purities then the other strategies (random and nonadaptive). Nonetheless, one should be aware of such variability of strategies at the end of the training.

To further analyze the variability of training, we focus on the state-preparation case from a thermal state, as in Fig. 5. In Fig. 15, the performance for many different target states is evaluated, along with the uncertainty due to different initial conditions.

2. Impact of the batch size on generalization

A final analysis that we conduct deals with the effect of the batch size during training. We want to analyze both the

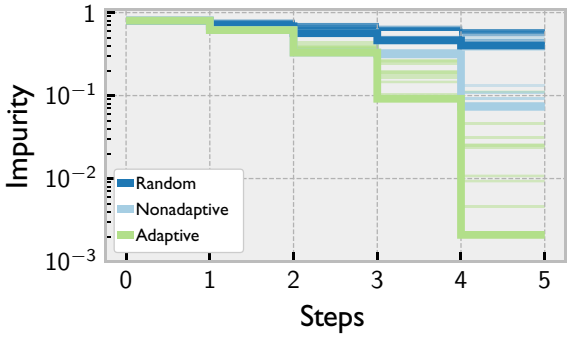

FIG. 14. The purification of a thermal state (with $\bar{n} = 2$) as in Fig. 4(c) of the main text. Here, the shaded lines show ten different strategies found by repeated runs of the algorithm, from different random starting points. The thick lines represent the best strategy found.

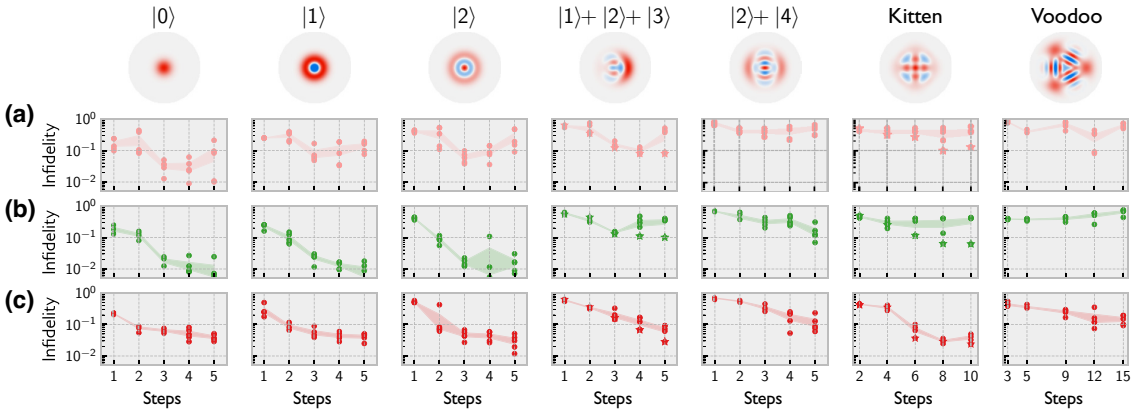

FIG. 15. State preparation from a thermal state ($\bar{n} = 1$), employing feedback, as in Fig. 5). We show the final infidelity as a function of the number of time steps available for the strategy. The columns represents various final target states, while the rows show the three different approaches to obtaining the control parameters: (a) an NN being fed the current quantum state as input; (b) a RNN obtaining the measurement sequence stepwise; (c) a look-up table as defined in the main text). Each dot represents a different training run with different initial conditions. For each number of steps, five runs are shown. Their mean and standard deviation are represented, respectively, as the center and width of the shaded area. For the kitten and the $|1\rangle + |2\rangle + |3\rangle$ state, we add as reference the data (plotted as stars) shown in Fig. 5.

performance during training and the generalization capabilities of the strategy learned. In order to assess that, we focus on the state-preparation case from a thermal state (Fig. 5). We carry out different training runs with distinct batch sizes (ranging from one to 100). For each batch size, we carry out five different training runs. We then postselect the best one, by computing the average fidelity on a much larger batch size (i.e., 1000). In Fig. 16 we then show the best-performing NN, RNN, or look-up table. Interestingly, even though the training is noisier with a lower batch size, it seems that feedback GRAPE can converge faster and to higher-fidelity solutions. Also, it seems that lower batch sizes can generalize well to higher ones. The better performance of lower batch sizes could be due to the possibility that the optimizer can escape local minima more efficiently than at larger batch sizes.

APPENDIX J: DETAILED ANALYSIS OF STRATEGIES DISCOVERED BY FEEDBACK GRAPE FOR THE JAYNES-CUMMINGS MODEL SCENARIO

In our work, we choose several different tasks within a Jaynes-Cummings model to illustrate the performance of our approach. Despite being only an illustrative physical

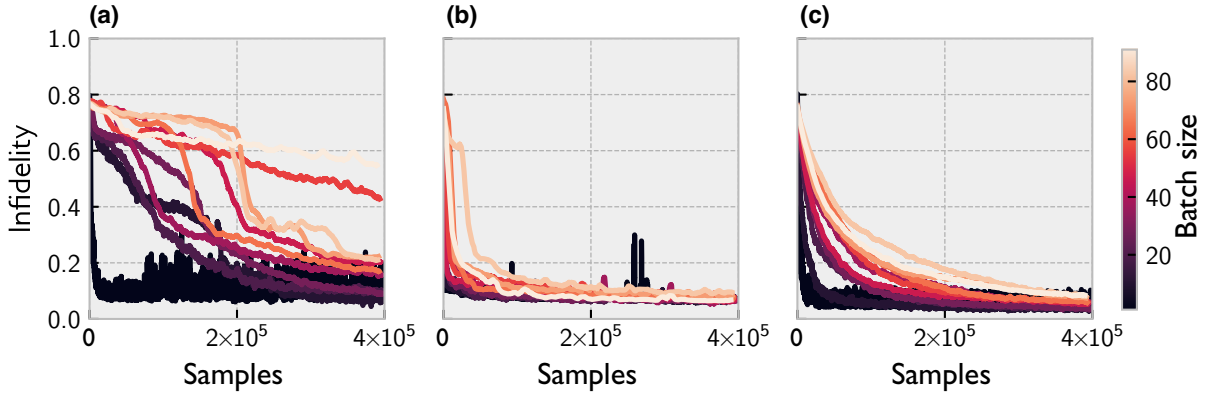

FIG. 16. State preparation from a thermal state (here, $\bar{n} = 1$) and for target state $\psi = (|1\rangle + |2\rangle + |3\rangle)/\sqrt{3}$, in six steps. Each training run is carried out with a different batch size (encoded in the color), and the performance of the resulting strategy—(a) NN, (b) RNN, and (c) look-up—is evaluated with a larger batch size (here, 1000) to suppress statistical noise. The training curves are plotted with a moving average of 50 trajectories in order to suppress fluctuations. The number of trajectories (i.e., batch size × number of gradient optimization steps) for each curve is the same.

example in this context, the model is of sufficient interest as a paradigm for actual feedback control of quantum optical systems. In this appendix, we describe some of the insights we are able to extract by closer inspection of the numerical results obtained by feedback GRAPE, in situations with feedback.

In the main text, we show the decision tree for the purification of a thermal state with initial occupation number $\langle \hat{a}^\dagger \hat{a} \rangle = 2$ in four measurements. Here, we want to show how the insight gained by analyzing the decision tree for this special case allows us to derive an analytical solution for an optimal purification strategy that is valid for an arbitrary temperature and number of measurements.

We start by reviewing the physics for the building-block measurement [cf. Eq. (4)]. This type of measurement has been originally proposed in Ref. [62] and has been used extensively in quantum optics experiments with flying Rydberg atoms, e.g., to monitor the occupation number of a cavity in the presence of very small thermal fluctuations [65] or to prepare a Fock state starting from an initial coherent state [64]. After each measurement, the Fock-state probability distribution $P_j(n)$ is updated by multiplying it with a sinusoidal mask,

$$P_{j+1}(n) \propto P_j(n) \cos^2\left[\gamma_i n + \frac{\delta_i}{2} + \pi(1 - m_j)/4\right]. \quad \text{(J1)}$$

To better understand the effects of the measurement, it is important to keep in mind two key insights. (i) If the measurement strength can be well approximated with a rational multiple of π, $\gamma_i = \pi p_i/q_i$, where p_i and q_i are coprime numbers, the denominator q_i represents the period of the mask. Thus, the relative occupations $P(n)/P(n')$ of any pair of Fock states that have the same excitation number modulus q_i, $(n - n')$ mod $q_i = 0$ do not change after the measurement. (ii) If the phase δ_i satisfies either condition

$$\pi \frac{p_i}{q_i} n_i + \frac{\delta_i}{2} = 0 \bmod \pi, \quad \text{or} \quad = \pi/2 \bmod \pi,$$

for an integer n_i, one measurement outcome ($m_i = -1$ or $m_i = 1$, respectively) rules out the infinite set of Fock states with excitation numbers n satisfying n mod $q_i = n_i$. We note that if q_i is an even number, any δ_i that satisfies the first condition for $n_i \equiv n_{i,-1}$ also satisfies the second condition for $n_i = n_{i,1} \equiv (n_{i,-1} + q_i/2)$ mod q_i. In this scenario, each of the two possible measurement outcomes rules out a (different) infinite set of Fock states, $n_{i,\pm 1}$ mod q_i for $m_i = \pm 1$. We note further that there are infinitely many values of δ_i satisfying one of the two conditions in Eq. (J1) for the same n_i. All of these values of δ_i are rational multiples of π.

Motivated by insights (i) and (ii), we write an algorithm that identifies values of γ_i and δ_i that are close to rational multiples of π with small denominators (we allow a deviation of 1% of π) and we display these rational values (in units of π) in the decision tree as shown in Fig. 4(d). By inspecting this decision tree, one can immediately observe that the NN tends to use measurement strength γ_j corresponding to the period $q_j = 2^j$ for the jth measurement. In order to understand this pattern, we inspect the phases δ_j selected by the NN. For the first measurement, the measurement strength is $\gamma_1 = \pi/2$ and the phase $\delta_1 = 0$. This corresponds to $n_{1,-1} = 0$ and $n_{1,1} = 1$. In other words, the Fock states 0 (1), along with all other even (odd) states, are ruled out by the measurement $m_1 = -1$ ($m_1 = 1$). Thus, the net effect is that, irrespective of the measurement outcome, the probability of every second Fock state is set to zero. Such a measurement extracts exactly 1 bit of information in the large-temperature limit. For the second measurement, the NN doubles the period of the sinusoidal mask, $q_2 = 4$ (independent of the outcome of the first measurement). By inspecting the phases δ_2 chosen adaptively by the NN, we find out that they always allow us to rule out either of the two most likely states after the measurement. For example, in the upper branch (corresponding to $m_1 = 1$), all odd states are decimated and, thus, the two more likely states are the 0 and 2 Fock states. From the tree, we see that $\delta_2 = \pi/2$ in this branch. This indeed satisfies the two conditions in Eq. (J1) with $n_i = n_{2,-1} = 2$ and $n_i = n_{2,1} = 0$, respectively. In other words, the Fock states with n mod $4 = 0$ (n mod $4 = 2$) are ruled out by the measurement outcome $m_2 = 1$ ($m_2 = -1$). Since all odd Fock states have already been ruled out after the first measurement, the overall effect of the first two measurements is to postselect every fourth Fock state, n mod $4 = 0$ (n mod $4 = 2$) for $m_1 = 1$ and $m_2 = -1$ ($m_1 = m_2 = 1$). Likewise, the choice of the phase $\delta_2 = -\pi/4$ in the lower branch allows us to postselect every fourth Fock state—now, n mod $4 = 1$ and n mod $4 = 3$ for $m_2 = 1$ and $m_2 = -1$, respectively. This strategy can be easily generalized for any arbitrarily large number of measurements J: the period q_i is doubled after every measurement, $q_j = 2^j$, independent of the measurement outcomes, and appropriate adaptive phases δ_j are selected to always rule out either of the two most likely states. Such a strategy allows us to postselect the Fock states with n mod $2^J = n_i$, where n_i depends on the measurement history. More precisely, there is a bijective mapping between $0 \le n_i < 2^J - 1$ and the 2^J possible measurement outcomes. Indeed, a close inspection of the strength γ_i and phases δ_i selected by the NN shows that the NN adopts this strategy for all four measurements in most (but not all) branches. A notable exception is the third measurement in the lowest branch (corresponding to $m_1 = m_2 = -1$). This choice results in an ineffective measurement that does not allow us to exclude either of the two most likely states. Interestingly, in this case the NN selects for the fourth measurement the measurement settings that were already expected (according to the strategy identified

above) for the third measurement. We believe that this suboptimal strategy corresponds to a local minimum for the gradient ascent. We note that the strategy the tree for which is displayed in Fig. 4(d) is obtained after selecting the best gradient-ascent training run out of ten runs with different random initializations. A tree without any such suboptimal measurements could be obtained by performing more gradient-ascent runs or, more efficiently, by increasing the temperature of the initial mixed state (which will punish more suboptimal purification strategies).

The same optimal strategy as discussed above can be implemented for infinitely many different choices of γ_j and δ_j. In particular, different bijective mappings between the measurement outcomes and the most likely state n_j after j measurements can be implemented. To find a simple analytical solution for the phases γ_j for one of the implementations of the optimal strategy, we choose $p_j = 1$ and, thus, $\gamma_j = \pi/2^j$. In addition, we choose n_j as the number the binary representation of which is $d_{j-1} \ldots d_2 d_1$ with $d_i = (1 - m_i)/2$, e.g., for $m_1 = m_2 = -1$ corresponding to $d_1 = d_2 = 1$ we have $n_3 = 1 + 2 = 3$. This mapping is implemented if the phase δ_j always allows us to rule out the Fock state with the largest probability (or, equivalently, the lowest excitation number among the states that have not yet been decimated by previous measurements) for the measurement outcome $m_j = -1$. With these constraints, we find a simple analytical solution for the phases, $\delta_j = \pi n_j/2^j$.

APPENDIX K: SYMMETRY OF THE OPTIMIZATION LANDSCAPE FOR SPIN-STATE PREPARATION WITH UNCERTAIN PARAMETERS

In this appendix, we analyze the symmetries of the optimization landscape for the learning of feedback strategies to prepare an ensemble of qubits in the excited state investigated in Sec. III F of the main text.

For $N = 2$, the 3D optimization landscape displayed as three cuts in Fig. 8(d) is the average $\langle\langle\mathcal{F}_2\rangle_{\mathbf{m}}\rangle_{g\sim P(g)}$ of the coupling-dependent fidelity

$$\langle\mathcal{F}_2\rangle_{\mathbf{m}} = \sin^2[g\tau_0/2]\cos^2[g\tau_1(m_0 = -1)/2] + \cos^2[g\tau_0/2]\sin^2[g\tau_1(m_0 = 1)/2]. \quad \text{(K1)}$$

We note that this function has three mirror planes because it is invariant under a sign change of τ_0, $\tau_1(m_0 = 1)$, or $\tau_1(m_0 = -1)$. In addition, in the plane $\tau_1(m_0 = -1) = 0$, corresponding to the leftmost cut in Fig. 8(d), it can be rewritten as

$$\langle\mathcal{F}_2\rangle_{\mathbf{m}} = \sin^2[g\tau_0/2] + \sin^2[g\tau_1(m_0 = 1)/2] - \sin^2[g\tau_0/2]\sin^2[g\tau_1(m_0 = 1)/2]. \quad \text{(K2)}$$

From the above expression, it becomes clear that one can exchange τ_0, $\tau_1(m_0 = 1)$ without changing the fidelity. These symmetries are present for any value of g and, thus, also for any weighted average over g and, in particular, for the optimization landscape $\langle\langle\mathcal{F}_2\rangle_{\mathbf{m}}\rangle_{g\sim P(g)}$. Since the optimal solutions lie on the plane $\tau_1(m_0 = -1) = 0$, there are eight symmetry-related optimal solutions corresponding to the same coupling-dependent fidelity $\langle\mathcal{F}\rangle_{\mathbf{m}}$.

This result can be generalized to the case of N measurements. On the hyperplane with $\tau_j(\mathbf{m}_{j-1}) = 0$ for all $\mathbf{m}_{j-1} \neq (1, \ldots, 1)$ the coupling-dependent fidelity is the function

$$\langle\mathcal{F}_N\rangle_{\mathbf{m}} = \sum_{j=0}^{N-1} \sin^2[g\tau_j/2] \prod_{j'=0}^{N-2} \cos^2[g\tau_{j'}/2], \quad \text{(K3)}$$

where $\tau_j = \tau_j(\mathbf{m}_{j-1})$, with $\mathbf{m}_{j-1} = (1, \ldots, 1)$. It is easy to show that this function is symmetric under permutation of its N variables τ_j. This leads to $4 \times N!$ optimal solutions with the same coupling-dependent fidelity $\langle\mathcal{F}\rangle_{\mathbf{m}}$.

[1] S. J. Glaser, U. Boscain, T. Calarco, C. P. Koch, W. Köckenberger, R. Kosloff, I. Kuprov, B. Luy, S. Schirmer, T. Schulte-Herbrüggen, D. Sugny, and F. K. Wilhelm, Training Schrödinger's cat: Quantum optimal control, Eur. Phys. J. D **69**, 279 (2015).

[2] C. P. Koch, Controlling open quantum systems: Tools, achievements, and limitations, J. Phys.: Condens. Matter **28**, 213001 (2016).

[3] N. Khaneja, T. Reiss, C. Kehlet, T. Schulte-Herbrüggen, and S. J. Glaser, Optimal control of coupled spin dynamics: Design of NMR pulse sequences by gradient ascent algorithms, J. Magn. Reson. **172**, 296 (2005).

[4] S. Machnes, U. Sander, S. J. Glaser, P. de Fouquières, A. Gruslys, S. Schirmer, and T. Schulte-Herbrüggen, Comparing, optimizing, and benchmarking quantum-control algorithms in a unifying programming framework, Phys. Rev. A **84**, 022305 (2011).

[5] F. Dolde, V. Bergholm, Y. Wang, I. Jakobi, B. Naydenov, S. Pezzagna, J. Meijer, F. Jelezko, P. Neumann, T. Schulte-Herbrüggen, J. Biamonte, and J. Wrachtrup, High-fidelity spin entanglement using optimal control, Nat. Commun. **5**, 3371 (2014).

[6] C. H. Yang, K. W. Chan, R. Harper, W. Huang, T. Evans, J. C. C. Hwang, B. Hensen, A. Laucht, T. Tanttu, F. E. Hudson, S. T. Flammia, K. M. Itoh, A. Morello, S. D. Bartlett, and A. S. Dzurak, Silicon qubit fidelities approaching incoherent noise limits via pulse engineering, Nat. Electron. **2**, 151 (2019).

[7] T. Schulte-Herbrüggen, A. Spörl, N. Khaneja, and S. J. Glaser, Optimal control-based efficient synthesis of building blocks of quantum algorithms: A perspective from network complexity towards time complexity, Phys. Rev. A **72**, 042331 (2005).

[8] A. Spörl, T. Schulte-Herbrüggen, S. J. Glaser, V. Bergholm, M. J. Storcz, J. Ferber, and F. K. Wilhelm, Optimal control of coupled Josephson qubits, Phys. Rev. A **75**, 012302 (2007).
[9] R. W. Heeres, P. Reinhold, N. Ofek, L. Frunzio, L. Jiang, M. H. Devoret, and R. J. Schoelkopf, Implementing a universal gate set on a logical qubit encoded in an oscillator, Nat. Commun. **8**, 94 (2017).
[10] R. Fisher, F. Helmer, S. J. Glaser, F. Marquardt, and T. Schulte-Herbrüggen, Optimal control of circuit quantum electrodynamics in one and two dimensions, Phys. Rev. B **81**, 085328 (2010).
[11] T. Schulte-Herbrüggen, A. Spörl, N. Khaneja, and S. J. Glaser, Optimal control for generating quantum gates in open dissipative systems, J. Phys. B: At., Mol. Opt. Phys. **44**, 154013 (2011).
[12] S. Boutin, C. K. Andersen, J. Venkatraman, A. J. Ferris, and A. Blais, Resonator reset in circuit QED by optimal control for large open quantum systems, Phys. Rev. A **96**, 042315 (2017).
[13] D. J. Egger and F. K. Wilhelm, Adaptive Hybrid Optimal Quantum Control for Imprecisely Characterized Systems, Phys. Rev. Lett. **112**, 240503 (2014).
[14] R.-B. Wu, B. Chu, D. H. Owens, and H. Rabitz, Data-driven gradient algorithm for high-precision quantum control, Phys. Rev. A **97**, 042122 (2018).
[15] R.-B. Wu, H. Ding, D. Dong, and X. Wang, Learning robust and high-precision quantum controls, Phys. Rev. A **99**, 042327 (2019).
[16] H.-J. Ding, B. Chu, B. Qi, and R.-B. Wu, Collaborative Learning of High-Precision Quantum Control and Tomography, Phys. Rev. Appl. **16**, 014056 (2021).
[17] P. de Fouquieres, S. G. Schirmer, S. J. Glaser, and I. Kuprov, Second order gradient ascent pulse engineering, J. Magn. Reson. **212**, 412 (2011).
[18] S. Machnes, E. Assémat, D. Tannor, and F. K. Wilhelm, Tunable, Flexible, and Efficient Optimization of Control Pulses for Practical Qubits, Phys. Rev. Lett. **120**, 150401 (2018).
[19] J. Zhang, Y.-x. Liu, R.-B. Wu, K. Jacobs, and F. Nori, Quantum feedback: Theory, experiments, and applications, Phys. Rep. Quantum Feedback: Theory, Exp. Appl. **679**, 1 (2017).
[20] R. v. Handel, J. K. Stockton, and H. Mabuchi, Modelling and feedback control design for quantum state preparation, J. Opt. B: Quantum Semiclass. Opt. **7**, S179 (2005).
[21] C. Sayrin, I. Dotsenko, X. Zhou, B. Peaudecerf, T. Rybarczyk, S. Gleyzes, P. Rouchon, M. Mirrahimi, H. Amini, M. Brune, J.-M. Raimond, and S. Haroche, Real-time quantum feedback prepares and stabilizes photon number states, Nature **477**, 73 (2011).
[22] R. Vijay, C. Macklin, D. H. Slichter, S. J. Weber, K. W. Murch, R. Naik, A. N. Korotkov, and I. Siddiqi, Stabilizing Rabi oscillations in a superconducting qubit using quantum feedback, Nature **490**, 77 (2012).
[23] M. Hirose and P. Cappellaro, Coherent feedback control of a single qubit in diamond, Nature **532**, 77 (2016).
[24] C. K. Andersen, A. Remm, S. Lazar, S. Krinner, J. Heinsoo, J.-C. Besse, M. Gabureac, A. Wallraff, and C. Eichler, Entanglement stabilization using ancilla-based parity detection and real-time feedback in superconducting circuits, npj Quantum Inf. **5**, 1 (2019).
[25] A. Hentschel and B. C. Sanders, Efficient Algorithm for Optimizing Adaptive Quantum Metrology Processes, Phys. Rev. Lett. **107**, 233601 (2011).
[26] C. Ahn, A. C. Doherty, and A. J. Landahl, Continuous quantum error correction via quantum feedback control, Phys. Rev. A **65**, 042301 (2002).
[27] J. Cramer, N. Kalb, M. A. Rol, B. Hensen, M. S. Blok, M. Markham, D. J. Twitchen, R. Hanson, and T. H. Taminiau, Repeated quantum error correction on a continuously encoded qubit by real-time feedback, Nat. Commun. **7**, 11526 (2016).
[28] C. Ryan-Anderson, J. G. Bohnet, K. Lee, D. Gresh, A. Hankin, J. P. Gaebler, D. Francois, A. Chernoguzov, D. Lucchetti, N. C. Brown, T. M. Gatterman, S. K. Halit, K. Gilmore, J. A. Gerber, B. Neyenhuis, D. Hayes, and R. P. Stutz, Realization of Real-Time Fault-Tolerant Quantum Error Correction, Phys. Rev. X **11**, 041058 (2021).
[29] S. Krinner, N. Lacroix, A. Remm, A. Di Paolo, E. Genois, C. Leroux, C. Hellings, S. Lazar, F. Swiadek, J. Herrmann, G. J. Norris, C. K. Andersen, M. Müller, A. Blais, C. Eichler, and A. Wallraff, (2021), ArXiv:2112.03708.
[30] R. S. Sutton and A. G. Barto, *Reinforcement Learning, Second Edition: An Introduction* (MIT Press, Cambridge, MA, 2018).
[31] M. Bukov, A. G. R. Day, D. Sels, P. Weinberg, A. Polkovnikov, and P. Mehta, Reinforcement Learning in Different Phases of Quantum Control, Phys. Rev. X **8**, 031086 (2018).
[32] M. August and J. M. Hernández-Lobato, in *High Performance Computing*, Lecture Notes in Computer Science, edited by R. Yokota, M. Weiland, J. Shalf, and S. Alam (Springer International Publishing, Cham, 2018), p. 591..
[33] M. Y. Niu, S. Boixo, V. N. Smelyanskiy, and H. Neven, Universal quantum control through deep reinforcement learning, npj Quantum Inf. **5**, 1 (2019).
[34] R. Porotti, D. Tamascelli, M. Restelli, and E. Prati, Coherent transport of quantum states by deep reinforcement learning, Commun. Phys. **2**, 1 (2019).
[35] Y. Baum, M. Amico, S. Howell, M. Hush, M. Liuzzi, P. Mundada, T. Merkh, A. R. Carvalho, and M. J. Biercuk, Experimental Deep Reinforcement Learning for Error-Robust Gate-Set Design on a Superconducting Quantum Computer, PRX Quantum **2**, 040324 (2021).
[36] T. Fösel, P. Tighineanu, T. Weiss, and F. Marquardt, Reinforcement Learning with Neural Networks for Quantum Feedback, Phys. Rev. X **8**, 031084 (2018).
[37] S. Borah, B. Sarma, M. Kewming, G. J. Milburn, and J. Twamley, Measurement-Based Feedback Quantum Control with Deep Reinforcement Learning for a Double-Well Nonlinear Potential, Phys. Rev. Lett. **127**, 190403 (2021).
[38] V. V. Sivak, A. Eickbusch, H. Liu, B. Royer, I. Tsioutsios, and M. H. Devoret, Model-Free Quantum Control with Reinforcement Learning, Phys. Rev. X **12**, 011059 (2022).
[39] R. Porotti, A. Essig, B. Huard, and F. Marquardt, (2021), ArXiv:2107.08816.

[40] K. Reuer, J. Landgraf, T. Fösel, J. O'Sullivan, L. Beltrán, A. Akin, G. J. Norris, A. Remm, M. Kerschbaum, and J.-C. Besse, *et al.*, (2022), arXiv preprint ArXiv:2210.16715.
[41] E. A. R. B. Sivak and V. V., *et al.*, Nature, 50 (2023).
[42] T. M. Moerland, J. Broekens, A. Plaat, and C. M. Jonker, Model-based reinforcement learning: A survey (2022), ArXiv:2006.16712.
[43] N. Leung, M. Abdelhafez, J. Koch, and D. Schuster, Speedup for quantum optimal control from automatic differentiation based on graphics processing units, Phys. Rev. A **95**, 042318 (2017).
[44] M. Abdelhafez, B. Baker, A. Gyenis, P. Mundada, A. A. Houck, D. Schuster, and J. Koch, Universal gates for protected superconducting qubits using optimal control, Phys. Rev. A **101**, 022321 (2020).
[45] F. Schäfer, M. Kloc, C. Bruder, and N. Lörch, A differentiable programming method for quantum control, Machine Learn.: Sci. Technol. **1**, 035009 (2020).
[46] H.-J. Liao, J.-G. Liu, L. Wang, and T. Xiang, Differentiable Programming Tensor Networks, Phys. Rev. X **9**, 031041 (2019).
[47] L. Coopmans, D. Luo, G. Kells, B. K. Clark, and J. Carrasquilla, Protocol Discovery for the Quantum Control of Majoranas by Differentiable Programming and Natural Evolution Strategies, PRX Quantum **2**, 020332 (2021).
[48] M. Abdelhafez, D. I. Schuster, and J. Koch, Gradient-based optimal control of open quantum systems using quantum trajectories and automatic differentiation, Phys. Rev. A **99**, 052327 (2019).
[49] H. M. Wiseman and G. J. Milburn, *Quantum Measurement and Control* (Cambridge University Press, Cambridge, United Kingdom, 2009).
[50] F. Schäfer, P. Sekatski, M. Koppenhöfer, C. Bruder, and M. Kloc, Control of stochastic quantum dynamics by differentiable programming, Machine Learn.: Sci. Technol. **2**, 035004 (2021).
[51] A. C. Doherty and K. Jacobs, Feedback control of quantum systems using continuous state estimation, Phys. Rev. A: At., Mol., Opt. Phys. **60**, 2700 (1999).
[52] A. Blais, A. L. Grimsmo, S. M. Girvin, and A. Wallraff, Circuit quantum electrodynamics, Rev. Mod. Phys. **93**, 025005 (2021).
[53] T. Q. Chen, Y. Rubanova, J. Bettencourt, and D. K. Duvenaud, in *32nd Conference on Neural Information Processing Systems* (Montreal, 2018), p. 13.
[54] M. Abadi, *et al.*, TensorFlow: Large-scale machine learning on heterogeneous systems (2015).
[55] S. Hochreiter and J. Schmidhuber, Long short-term memory, Comparison of quantum and semiclassical radiation theories with applications to the beam maser, Neural Comput. **9**, 1735 (1997).
[56] E. Jaynes and F. Cummings, Proc. IEEE **51**, 89 (1963).
[57] B. W. Shore and P. L. Knight, The Jaynes-Cummings model, J. Mod. Opt. **40**, 1195 (1993).
[58] C. K. Law and J. H. Eberly, Arbitrary Control of a Quantum Electromagnetic Field, Phys. Rev. Lett. **76**, 1055 (1996).
[59] M. Hofheinz, H. Wang, M. Ansmann, R. C. Bialczak, E. Lucero, M. Neeley, A. D. O'Connell, D. Sank, J. Wenner, J. M. Martinis, and A. N. Cleland, Synthesizing arbitrary quantum states in a superconducting resonator, Nature **459**, 546 (2009).
[60] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, Proximal policy optimization algorithms (2017), ArXiv:1707.06347.
[61] D. Gottesman, A. Kitaev, and J. Preskill, Encoding a qubit in an oscillator, Phys. Rev. A **64**, 012310 (2001).
[62] M. Brune, S. Haroche, V. Lefevre, J. M. Raimond, and N. Zagury, Quantum Nondemolition Measurement of Small Photon Numbers by Rydberg-Atom Phase-Sensitive Detection, Phys. Rev. Lett. **65**, 976 (1990).
[63] M. Brune, E. Hagley, J. Dreyer, X. Maître, A. Maali, C. Wunderlich, J. M. Raimond, and S. Haroche, Observing the Progressive Decoherence of the "Meter" in a Quantum Measurement, Phys. Rev. Lett. **77**, 4887 (1996).
[64] C. Guerlin, J. Bernu, S. Deléglise, C. Sayrin, S. Gleyzes, S. Kuhr, M. Brune, J.-M. Raimond, and S. Haroche, Progressive field-state collapse and quantum non-demolition photon counting, Nature **448**, 889 (2007).
[65] S. Gleyzes, S. Kuhr, C. Guerlin, J. Bernu, S. Deléglise, U. Busk Hoff, M. Brune, J.-M. Raimond, and S. Haroche, Quantum jumps of light recording the birth and death of a photon in a cavity, Nature **446**, 297 (2007).
[66] S. Krastanov, V. V. Albert, C. Shen, C.-L. Zou, R. W. Heeres, B. Vlastakis, R. J. Schoelkopf, and L. Jiang, Universal control of an oscillator with dispersive coupling to a qubit, Phys. Rev. A **92**, 040303 (2015).
[67] J. Degrave, *et al.*, Magnetic control of tokamak plasmas through deep reinforcement learning, Nature **602**, 414 (2022).
[68] Z. Xie, Z. Lin, J. Li, S. Li, and D. Ye, Pretraining in deep reinforcement learning: A survey (2022), ArXiv:2211.03959.
[69] J. Yao, L. Lin, and M. Bukov, Reinforcement Learning for Many-Body Ground-State Preparation Inspired by Counterdiabatic Driving, Phys. Rev. X **11**, 031070 (2021).
[70] D. Wierichs, C. Gogolin, and M. Kastoryano, Avoiding local minima in variational quantum eigensolvers with the natural gradient optimizer, Phys. Rev. Res. **2**, 043246 (2020).
[71] J. R. McClean, S. Boixo, V. N. Smelyanskiy, R. Babbush, and H. Neven, Barren plateaus in quantum neural network training landscapes, Nat. Commun. **9**, 4812 (2018).
[72] E. Grant, L. Wossnig, M. Ostaszewski, and M. Benedetti, An initialization strategy for addressing barren plateaus in parametrized quantum circuits, Quantum **3**, 214 (2019).
[73] T. Volkoff and P. J. Coles, Large gradients via correlation in random parameterized quantum circuits, Quantum Sci. Technol. **6**, 025008 (2021).
[74] Z. Holmes, K. Sharma, M. Cerezo, and P. J. Coles, Connecting Ansatz Expressibility to Gradient Magnitudes and Barren Plateaus, PRX Quantum **3**, 010313 (2022).
[75] S. Khatri, R. LaRose, A. Poremba, L. Cincio, A. T. Sornborger, and P. J. Coles, Quantum-assisted quantum compiling, Quantum **3**, 140 (2019).
[76] M. Cerezo, A. Sone, T. Volkoff, L. Cincio, and P. J. Coles, Cost function dependent barren plateaus in shallow parametrized quantum circuits, Nat. Commun. **12**, 1791 (2021).

[77] R. Wiersema, C. Zhou, J. F. Carrasquilla, and Y. B. Kim, Measurement-induced entanglement phase transitions in variational quantum circuits (2021), ArXiv:2111.08035.

[78] A. Mousavian, A. Toshev, M. Fiser, J. Kosecka, A. Wahid, and J. Davidson, (2019), ArXiv:1805.06066.

[79] K. Kim, M. W. Lee, Y. Kim, J.-H. Ryu, M. Lee, and B.-T. Zhang, (2021), ArXiv:2110.12985.

[80] C. N. Self, K. E. Khosla, A. W. R. Smith, F. Sauvage, P. D. Haynes, J. Knolle, F. Mintert, and M. S. Kim, Variational quantum algorithm with information sharing, npj Quantum Inf. **7**, 1 (2021).

[81] A. Hill, A. Raffin, M. Ernestus, A. Gleave, A. Kanervisto, R. Traore, P. Dhariwal, C. Hesse, O. Klimov, A. Nichol, M. Plappert, A. Radford, J. Schulman, S. Sidor, and Y. Wu, Stable baselines (2018).

[82] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. P. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, Asynchronous methods for deep reinforcement learning (2016), ArXiv:1602.01783.

[83] M. Andrychowicz, F. Wolski, A. Ray, J. Schneider, R. Fong, P. Welinder, B. McGrew, J. Tobin, P. Abbeel, and W. Zaremba, Hindsight experience replay (2018), ArXiv:1707.01495.

[84] J. Schulman, S. Levine, P. Moritz, M. I. Jordan, and P. Abbeel, Trust region policy optimization (2017), ArXiv:1502.05477.

[85] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, Continuous control with deep reinforcement learning (2019), ArXiv:1509.02971.

[86] D. P. Kingma and J. Ba, ADAM: A method for stochastic optimization (2017), ArXiv:1412.6980.